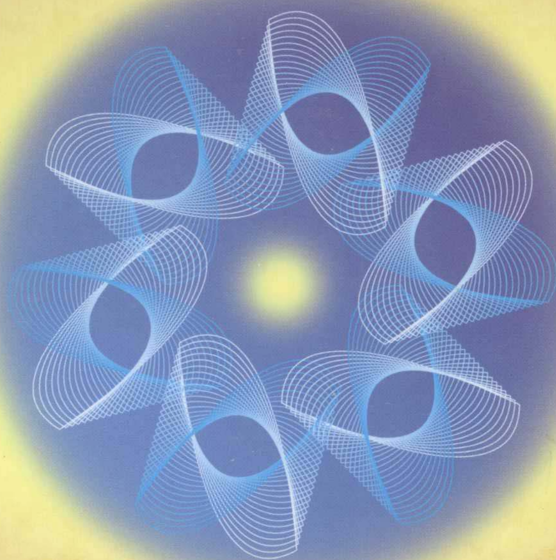
TURING 图灵数学·统计学丛书 52



# Matrix Computations

# 矩阵计算

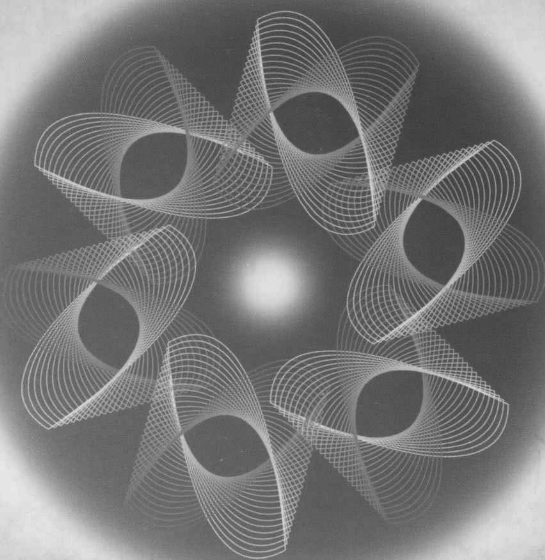
（第3版）


[美] Gene H. Golub 著
Charles F. Van Loan

袁亚湘等 译

人民邮电出版社
POSTS & TELECOM PRESS

图灵数学·统计学丛书 52



# Matrix Computations

# 矩阵计算

（第3版）

[美] Gene H. Golub
Charles F. Van Loan 著
袁亚湘等 译

人民邮电出版社
POSTS & TELECOM PRESS

**图书在版编目(CIP)数据**

矩阵计算：第3版/（美）戈卢布(Golub, G. H.),
（美）范洛恩（Van Loan, C. F.）著；袁亚湘等译. —北
京：人民邮电出版社, 2011.3
(图灵数学·统计学丛书)
书名原文：Matrix Computations
ISBN 978-7-115-24785-8

Ⅰ. ①矩… Ⅱ. ①戈… ②范… ③袁… Ⅲ. ①矩阵-
计算方法 Ⅳ. ①O241.6

中国版本图书馆 CIP 数据核字(2011)第 008044 号

## 内 容 提 要

本书是数值计算领域的名著，系统地介绍了矩阵计算的基本理论和方法. 内容包括：矩阵乘法、矩阵分析、线性方程组、正交化和最小二乘法、特征值问题、Lanczos 方法、矩阵函数及专题讨论等. 书中的许多算法都由现成的软件包来实现，每节后还附有习题，并有注释和大量参考文献.

本书可作为高等学校数学系高年级本科生和研究生教材，亦可作为计算数学和工程技术人员的参考用书.

图灵数学·统计学丛书

**矩 阵 计 算（第3版）**

◆ 著　　[美] Gene H. Golub　Charles F. Van Loan
　译　　袁亚湘等
　责任编辑　明永玲
◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
　邮编 100061　电子函件 315@ptpress.com.cn
　网址 http://www.ptpress.com.cn
　北京隆昌伟业印刷有限公司印刷
◆ 开本：700×1000 1/16
　印张：37
　字数：726 千字　　2011 年 3 月第 1 版
　印数：1–3 000 册　　2011 年 3 月北京第 1 次印刷

著作权合同登记号 图字：01-2008-5835 号

ISBN 978-7-115-24785-8

定价：89.00 元

**读者服务热线：(010)51095186　印装质量热线：(010) 67129223**

**反盗版热线：(010)67171154**

# 版 权 声 明

*Matrix Computations, Third Edition* by Gene H. Golub, Charles F. Van Loan, ISBN 0-8018-5414-8.

# 译 者 序

这本《矩阵计算》是数值线性代数方面最权威、最全面的专著, 已经成为计算数学的经典著作之一. 该书英文原著最早出版于 1983 年, 1989 年再版, 本书译自 1996 年的第 3 版. 原著作者之一是已故的 Gene H. Golub 教授 (1932—2007), 他生前是美国斯坦福大学教授, 美国科学院院士, 美国工程院院士, 国际上著名的计算数学家. Golub 先生对我国计算数学的发展十分关心, 也给予过大力的支持和帮助. 他曾多次来访中国, 特别值得提到的是, Golub 先生于 2000 年特为《矩阵计算》的中译本作序.

该书的翻译是由我和我们所的三位研究生聂家旺、叶军涛和朱宗武合作于 1999 年完成的, 中译本于 2001 年在科学出版社出版. 该中译本出版后受到广大读者的喜爱, 不少高校将该书选为教材或教学参考书.

2009 年, 人民邮电出版社图灵公司的编辑邀请我重新翻译《矩阵计算》一书. 征求了科学出版社同意后, 我在 2001 年出版的中译本基础上加以修订, 由人民邮电出版社重新出版.

在修订过程中, 图灵公司的编辑进行了认真细致的核对工作, 提出了不少很好的修改意见. 我对他们这种认真负责的工作态度表示敬意, 也对其提出的宝贵意见表示感谢.

我相信, 该书的再次出版, 将让更多的读者, 特别是年轻的读者, 对这本名著有所认识并从中受益, 这将对我国计算数学的发展和人才培养起到积极的作用.

# 中译本前言

我很高兴我们的著作被译成中文出版. 计算数学在当代中国正有着很大的发展. 这些成就是在已故的冯康教授的领导下取得的, 他的工作是开创性的, 为这一学科的发展播下了种子. 数值代数至关重要, 毫无疑问在中国已经得到蓬勃发展. 中国高等院校坚实的数学训练, 造就了许多优秀的青年数值分析专家, 这使美国和其他西方国家受益匪浅.

我希望中译本能进一步增强中国读者对该领域的兴趣, 并加强我们彼此间在该领域的联系与合作.

Gene H. Golub

2000.9.14

# 第 3 版前言

矩阵计算领域在不断发展和成熟. 在第 3 版中, 我们增加了 300 多篇新的文献和 100 多个新的问题. 对 LINPACK 和 EISPACK 的引述被替换为相应的 LAPACK 重要例程, 并在相应章首列表给出.

在第 1 版和第 2 版中, 我们列出了少量的综合性参考文献：Wilkinson(1965), Forsythe and Moler(1967), Stewart(1973), Hanson and Lawson(1974) 以及 Parlett (1980). 这些著作在研究领域仍然重要, 但也有一些相当好的新教科书和专著. 见之后的文献介绍.

与前两版一样, 我们在每一节之后都给出参考文献.

前两个版本有许多打印错误, 我们感谢读者向我们指出这些打印错误, 在新版中我们作了许多改正和澄清.

下面介绍新版的要点. 第 1 章 (矩阵乘法) 和第 6 章 (并行矩阵计算) 彻底重写了, 写作没有原来那样正式. 我们认为这样有利于培养对高性能计算的直觉, 将算法与实现更好地挂钩.

在第 2 章 (矩阵分析), 我们扩充了 CS 分解的处理并增加了一个证明. 按最新的发展状态, 更新了浮点数运算的内容. 在第 4 章 (特殊线性方程组)Toeplitz 矩阵那一节, 我们添加了与循环矩阵的关系以及快速傅里叶变换. 在讨论非定方程组时增加了关于平衡方程组的一节.

第 5 章 (正交化和最小二乘法) 给出了修正 Gram-Schmidt 方法的更精确的描述. 第 8 章 (对称特征值问题) 很大程度上进行了重写及重组, 以便尽可能减少它对第 7 章 (非对称特征值问题) 的依赖. 事实上, 现在这两章的关系已经小到可以随意先读其中之一.

第 9 章 (Lanczos 方法) 扩展了对非对称 Lanczos 方法以及 Arnoldi 迭代的讨论. 第 10 章 (线性方程组的迭代解法) 的“非对称部分”同样扩充了崭新的一节, 讨论各种 Krylov 子空间方法, 这些方法可用来处理稀疏非对称线性方程组求解.

在 12.5 节 (修正正交分解) 我们增加了关于 ULV 修正的一节. 在 12.6 节中讨论了 Toeplitz 矩阵特征值问题和正交矩阵特征值问题.

我们两人期待着与读者们继续对话. 正如我们在第 2 版前言中所说:“与如此有趣的和友好的读者打交道是很愉快的.”

许多人对第 3 版提出了有价值的建议. 特别要致谢的有 Greg Ammar, Mike Heath, Nick Trefethen 和 Steve Vavasis.

最后, 我们想感谢康奈尔大学的 Cindy Robinson. 一个专业的助手是非常重要的.

# 软　件

## LAPACK

书中的许多算法都在 LAPACK 软件包中实现:

E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuGroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen(1995). *LAPACK Users' Guide, Release 2.0, 2nd ed.*, SIAM Publications, Philadelphia.

这个软件包中一些重要例程之索引在下面几章的开头列出:

第 1 章　1 级, 2 级, 3 级 BLAS

第 3 章　一般线性方程组

第 4 章　特殊线性方程组

第 5 章　正交化和最小二乘法

第 7 章　非对称特征值问题

第 8 章　对称特征值问题

我们关于 LAPACK 的介绍不够详细, 但应该能够“让你入门”. 所以, 当我们说_TRSV可用来解三角方程组 $\boldsymbol{Ax}=\boldsymbol{b}$, 就是请你通过 LAPACK 手册去发现 $\boldsymbol{A}$ 可以是上三角形的或下三角形的, 而且也可处理转置方程组 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{b}$. 还有, 下划线是计算类型 (单精度、双精度、复数等) 的标识.

LAPACK 是基于其他两个在软件开发史上有划时代意义的软件包而开发的. EISPACK 是 20 世纪 70 年代初期开发的, 它用于求解对称、非对称及一般特征值问题.

B. T. Smith, J. M. Boyle, Y. Ikebe, V. C. Klema, and C. B. Moler(1970). *Matrix Eigensystem Routines*: *Eispack Guide, 2nd ed.*, Lecture Notes in Computer Science, Volume 6, Springer-Verlag, New York.

B. S. Garbow, J. M. Boyle, J. J. Dongarra, and C. B. Moler(1972). *Matrix Eigensystem Routines*: *Eispack Guide Extension*, Lecture Notes in Computer Science, Volume 51, Springer-Verlag, New York.

LINPACK 是 20 世纪 70 年代后期开发的求解线性方程组和最小二乘法问题的软件包.

EISPACK 和 LINPACK 基于一批关于主要的矩阵分解之 Algol 实现的文章. 这些文章收集在:

J. H. Wilkinson and C. Reinsch, eds. (1971). *Handbook for Automatic Computation, Vol 2., Linear Algebra*, Springer-Verlag, New York.

## NETLIB

许多软件, 包括 LAPACK, EISPACK 和 LINPACK 都可通过 Netlib 从网上得到:

www: http://www.netlib.org/index.html

ftp: ftp://ftp.netlib.org

也可通过 E-mail 发一行消息至:

mail netlib@ornl.gov

send index

来了解软件.

## MATLAB

MATLAB 实现了 LAPACK 并且给出了很流行的矩阵计算平台:

MATLAB *User's Guide*, The Math Works Inc., Natick, Massachusetts.

M. Marcus (1993). *Matrices and* MATLAB: *A Tutorial*, Prentice Hall, Upper Saddle River, NJ.

R. Pratap (1995). *Getting Started with* MATLAB, Saunders College Publishing, Fort Worth, TX.

本书中的许多习题最好在布置给学生时要求使用 MATLAB 解答. 在描述算法时, 我们广泛应用 MATLAB 记号.

# 精选参考文献

书中每一节都以带注释的参考文献结束. 以下列出的参考书总体覆盖了这个领域. 引用文献按以下方式分类:

| | |
|---|---|
| 1970 年之前的经典著作 | 开拓该领域的早期著作 |
| 概论入门 | 适合于本科生课堂学习 |
| 高级概论 | 最好给实习者和研究生 |
| 分析 | 数学基础 |
| 线性方程组问题 | $\boldsymbol{Ax} = \boldsymbol{b}$ |
| 线性拟合问题 | $\boldsymbol{Ax} \approx \boldsymbol{b}$ |
| 特征值问题 | $\boldsymbol{Ax} = \lambda \boldsymbol{x}$ |
| 高性能计算 | 并行/向量问题 |
| 文集 | 有用的专题文集 |

在每一类中都按出版年代为序排列.

### 1970 年之前的经典著作


V. N. Faddeeva(1959). *Computational Methods of Linear Algebra*, Dover, New York.

E. Bodewig(1959). *Matrix Calculus*, North Holland, Amsterdam.

R. S. Varga (1962). *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ.

J. H. Wilkinson (1963). *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, NJ.

A. S. Householder (1964). *Theory of Matrices in Numerical Analysis*, Blaisdell, New York. Reprinted in 1974 by Dover, New York.

L. Fox(1964). *An Introduction to Numerical Linear Algebra*, Oxford University Press, Oxford, England.

J. H. Wilkinson(1965). *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, England.

G. E. Forsythe and C. Moler(1967). *Computer Solution of Linear Algebraic Systems*, Prentice-Hall, Englewood Cliffs, NJ.


### 概论入门


A. R. Gourlay and G. A. Watson (1973). *Computational Methods for Matrix Eigenproblems*, John Wiley & Sons, New York.

G. W. Stewart(1973). *Introduction to Matrix Computations*, Academic Press, New York.

R. J. Goult, R. F. Hoskins, J. A. Milner and M. J. Pratt (1974). *Computational Methods in Linear Algebra*, John Wiley and Sons, New York.

T. F. Coleman and C. F. Van Loan(1988). *Handbook for Matrix Computations*, SIAM Publications, Philadelphia, PA.

W. W. Hager(1988). *Applied Numerical Linear Algebra*, Prentice-Hall, Englewood Cliffs, NJ.

P. G. Ciarlet (1989). *Introduction to Numerical Linear Algebra and Optimisation*, Cambridge University Press.

D. S. Watkins (1991). *Fundamentals of Matrix Computations*, John Wiley and Sons, New York.

P. Gill, W. Murray, and M. H. Wright (1991). *Numerical Linear Algebra and Optimization, Vol. 1*, Addison-Wesley, Reading, MA.

A. Jennings and J. J. McKeowen(1992). *Matrix Computation(2nd ed)*, John Wiley and Sons, New York.

B. N. Datta(1995). *Numerical Linear Algebra and Applications.* Brooks/Cole Publishing Company, Pacific Grove, California.

M. T. Heath (1997). *Scientific Computing: An Introductory Survey*, McGraw-Hill, New York.

C. F. Van Loan(1997). *Introduction to Scientific Computing: A Matrix-Vector Approach Using Matlab*, Prentice Hall, Upper Saddle River, NJ.

## 高级概论

N. J. Higham(1996). *Accuracy and Stability of Numerical Algorithms*, SIAM Publications, Philadelphia, PA.

J. W. Demmel(1996). *Numerical Linear Algebra*, SIAM Publications, Philadelphia, PA.

L. N. Trefethen and D. Bau III (1997). *Numerical Linear Algebra*, SIAM Publications, Philadelphia, PA.

## 分　析

F. R. Gantmacher(1959). *The Theory of Matrices Vol. 1*, Chelsea, New York.

F. R. Gantmacher(1959). *The Theory of Matrices Vol. 2*, Chelsea, New York.

A. Berman and R. J. Plemmons(1979). *Nonnegative Matrices in the Mathematical*

*Sciences*, Academic Press, New York. Reprinted with additions in 1994 by SIAM Publications, Philadelphia, PA.

G. W. Stewart and J. Sun(1990). *Matrix Perturbation Theory*, Academic Press, San Diego.

R. Horn and C. Johnson(1985). *Matrix Analysis*, Cambridge University Press, New York.

R. Horn and C. Johnson(1991). *Topics in Matrix Analysis*, Cambridge University Press, New York.

## 线性方程组问题

D. M. Young (1971). *Iterative Solution of Large Linear Systems*, Academic Press, New York.

L. A. Hageman and D. M. Young(1981). *Applied Iterative Methods*, Academic Press, New York.

A. George and J. W-H. Liu(1981). *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall Inc., Englewood Cliffs, New Jersey.

S. Pissanetsky(1984). *Sparse Matrix Technology*, Academic Press, New York.

I. S. Duff, A. M. Erisman, and J. K. Reid(1986). *Direct Methods for Sparse Matrices*, Oxford University Press, New York.

R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. van der Vorst(1993). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM Publications, Philadelphia, PA.

W. Hackbusch (1994). *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, New York.

O. Axelsson(1994). *Iterative Solution Methods*, Cambridge University Press.

Y. Saad(1996). *Iterative Methods for Sparse Linear Systems*, PWS Publishing Co., Boston.

## 线性拟合问题

C. L. Lawson and R. J. Hanson(1974). *Solving Least Squares Problems*, Prentice-Hall, Englewood, Cliffs, NJ.Reprinted with a detailed"new developments"appendix in 1996 by SIAM Publications, Philadelphia, PA.

R. W. Farebrother(1987). *Linear Least Squares Computations*, Marcel Dekker, New York.

S. Van Huffel and J. Vandewalle(1991). *The Total Least Squares Problem: Computational Aspects and Analysis*, SIAM Publications, Philadelphia, PA.

Å. Björck(1996). *Numerical Methods for Least Squares Problems*, SIAM Publications, Philadelphia, PA.

## 特征值问题

B. N. Parlett(1980). *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ.

J. Cullum and R. A. Willoughby(1985a). *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I Theory*, Birkhaüser, Boston.

J. Cullum and R. A. Willoughby(1985b). *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. II Programs*, Birkhaüser, Boston.

Y. Saad(1992). *Numerical Methods for Large Eigenvalue Problems: Theory and Algorithms*, John Wiley and Sons, New York.

F. Chatelin(1993). *Eigenvalues of Matrices*, John Wiley and Sons, New York.

## 高性能计算

W. Schönauer(1987). *Scientific Computing on Vector Computers*, North Holland, Amsterdam.

R. W. Hockney and C.R.Jesshope(1988). *Parallel Computers* 2, Adam Hilger, Bristol and Philadelphia.

J. J. Modi(1988). *Parallel Algorithms and Matrix Computation*, Oxford University Press, Oxford.

J. Ortega(1988). *Introduction to Parallel and Vector Solution of Linear Systems*, Plenum Press, New York.

J. Dongarra, I. Duff, D. Sorensen, and H. van der Vorst(1990). *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM Publications, Philadelphia, PA.

Y. Robert(1990). *The Impact of Vector and Parallel Architectures on the Gaussian Elimination Algorithm*, Halsted Press, New York.

G. H. Golub and J. M. Ortega(1993). *Scientific Computing: An Introduction with Parallel Computing*, Academic Press, Boston.

## 文　集

D. J. Rose and R. A. Willoughby, eds.(1972). *Sparse Matrices and Their Applica-*

*tions*, Plenum Press, New York.

J. R. Bunch and D. J. Rose, eds.(1976). *Sparse Matrix Computations*, Academic Press, New York.

I. S. Duff and G. W. Stewart, eds.(1979). *Sparse Matrix Proceedings*, 1978, SIAM Publications, Philadelphia, PA.

I. S. Duff, ed.(1981). *Sparse Matrices and Their Uses*, Academic Press, New York.

Å. Björck, R. J. Plemmons, and H. Schneider, eds.(1981). *Large-Scale Matrix Problems*, North-Holland, New York.

G. Rodrigue, ed.(1982). *Parallel Computation*, Academic Press, New York.

B. Kågström and A. Ruhe, eds.(1983). *Matrix Pencils*, Proc. Pite Havsbad, 1982, Lecture Notes in Mathematics 973, Springer-Verlag, New York and Berlin.

J. Cullum and R. A. Willoughby,eds.(1986). *Large Scale Eigenvalue Problems*, North-Holland, Amsterdam.

A. Wouk, ed. (1986). *New Computing Environments: Parallel, Vector, and Systolic, SIAM Publications*, Philadelphia, PA.

M. T. Heath, ed.(1986). *Proceedings of First SIAM Conference on Hypercube Multiprocessors*, SIAM Publications, Philadelphia, PA.

M. T. Heath, ed.(1987). *Hypercube Multiprocessors*, SIAM Publications, Philadelphia, PA.

G. Fox, ed.(1988). *The Third Conference on Hypercube Concurrent Computers and Applications, Vol. II-Applications*, ACM Press, New York.

M. H. Schultz, ed.(1988). *Numerical Algorithms for Modern Parallel Computer Architectures*, IMA Volumes in Mathematics and Its Applications, Number 13, Springer-Verlag, Berlin.

E. F. Deprettere, ed.(1988). *SVD and Signal Processing*. Elsevier, Amsterdam.

B. N. Datta, C. R. Johnson, M. A. Kaashoek, R. Plemmons, and E. D. Sontag, eds.(1988), *Linear Algebra in Signals, Systems, and Control*, SIAM Publications, Philadelphia, PA.

J. Dongarra, I. Duff, P. Gaffney, and S. McKee, eds.(1989), *Vector and Parallel Computing*, Ellis Horwood, Chichester, England.

O. Axelsson, ed.(1989). "Preconditioned Conjugate Gradient Methods," *BIT* 29: 4.

K. Gallivan, M. Heath, E. Ng, J. Ortega, B. Peyton, R. Plemmons, C. Romine, A. Sameh, and B. Voigt(1990). *Parallel Algorithms for Matrix Computations*, SIAM Publications, Philadelphia, PA.

G. H. Golub and P. Van Dooren, eds.(1991). *Numerical Linear Algebra, Digital*

*Signal Processing, and Parallel Algorithms.* Springer-Verlag, Berlin.

R. Vaccaro, ed.(1991). *SVD and Signal Processing II: Algorithms, Analysis, and Applications.* Elsevier, Amsterdam.

R. Beauwens and P. de Groen, eds. (1992). *Iterative Methods in Linear Algebra,* Elsevier(North-Holland), Amsterdam.

R. J. Plemmons and C. D. Meyer, eds. (1993). *Linear Algebra, Markov Chains, and Queuing Models,* Springer-Verlag, New York.

M. S. Moonen, G. H. Golub, and B. L. R. de Moor, eds.(1993). *Linear Algebra for Large Scale and Real-Time Applications,* Kluwer, Dordrecht, The Netherlands.

J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, eds.(1994). *Proceedings of the Cornelius Lanczos International Centenary Conference,* SIAM Publications Philadelphia, PA.

R. V. Patel, A. J. Laub, and P. M. Van Dooren, eds.(1994). *Numerical Linear Algebra Techniques for Systems and Control,* IEEE Press, Piscataway, New Jersey.

J. Lewis, ed. (1994). *Proceedings of the Fifth SIAM Conference on Applied Linear Algebra,* SIAM Publications, Philadelphia, PA.

A. Bojanczyk and G. Cybenko, eds.(1995). *Linear Algebra for Signal Processing,* IMA Volumes in Mathematics and Its Applications, Springer-Verlag, New York.

M. Moonen and B. De Moor, eds.(1995). *SVD and Signal Processing III: Algorithms, Analysis, and Applications.* Elsevier, Amsterdam.

# 目　录

# 第1章 矩阵乘法

研究矩阵计算的合适出发点是矩阵与矩阵的乘法. 这一问题在数学上虽然简单, 但从计算上来看却是十分丰富的. 在 1.1 节中, 我们将看到矩阵相乘可以有好几种不同的形式. 还将引入矩阵划分的概念, 并将其用来刻画计算上的几种线性代数的“级”.

如果一个矩阵具有某种结构, 则它常常可加以利用. 例如, 一个对称矩阵只需一个一般矩阵的一半空间即可储存. 在矩阵乘向量中, 如果矩阵中有许多零元素, 则可减少许多计算时间. 这些问题将在 1.2 节中讨论.

在 1.3 节中定义了分块矩阵记号. 分块矩阵是一个以矩阵为元素的矩阵. 这一概念无论是在理论上还是在实践中都是十分重要的. 在理论方面, 分块矩阵记号使得重要的矩阵分解的证明十分简洁. 这些分解是数值线性代数的基石. 从计算的角度看, 分块算法中含有大量的高性能计算机结构所擅长的矩阵运算, 因而在矩阵乘法中是重要的.

这些新的结构要求算法设计者对流量与实际的计算量同等重视. 科学计算的这一特性在 1.4 节中阐明. 在该节还将讨论向量流水线计算的重要因素: 间、向量长度、向量存取的次数和向量再利用的程度.

*预备知识*

熟悉 MATLAB语言是重要的, 可参阅 Pratap(1995) 和 Van Loan(1996) 的教材. 更多介绍高性能矩阵计算的一本入门书是Dongarra, Duff, Sorensen, and Vorst(1991). 本章中与 LAPACK 相关的例程如下.

| LAPACK: 一些一般运算 | | |
|---|---|---|
| _SCAL | $\boldsymbol{x} \leftarrow \alpha \boldsymbol{x}$ | 标量与向量相乘 |
| _DOT | $\mu \leftarrow \boldsymbol{x}^{\mathrm{T}} \boldsymbol{y}$ | 点积 |
| _AXPY | $\boldsymbol{y} \leftarrow \alpha \boldsymbol{x} + \boldsymbol{y}$ | SAXPY |
| _GEMV | $\boldsymbol{y} \leftarrow \alpha \boldsymbol{A} \boldsymbol{x} + \beta \boldsymbol{y}$ | 矩阵与向量相乘 |
| _GER | $\boldsymbol{A} \leftarrow \boldsymbol{A} + \alpha \boldsymbol{x} \boldsymbol{y}^{\mathrm{T}}$ | 秩 1 修正 |
| _GEMM | $\boldsymbol{C} \leftarrow \alpha \boldsymbol{A} \boldsymbol{B} + \beta \boldsymbol{C}$ | 矩阵相乘 |

| LAPACK: 一些对称运算 | | |
|---|---|---|
| _SYMV | $\boldsymbol{y} \leftarrow \alpha \boldsymbol{A} \boldsymbol{x} + \beta \boldsymbol{y}$ | 矩阵与向量相乘 |
| _SPMV | $\boldsymbol{y} \leftarrow \alpha \boldsymbol{A} \boldsymbol{x} + \beta \boldsymbol{y}$ | 矩阵与向量相乘 |
| _SYR | $\boldsymbol{A} \leftarrow \alpha \boldsymbol{x} \boldsymbol{x}^{\mathrm{T}} + \boldsymbol{A}$ | 秩 1 修正 |
| _SYR2 | $\boldsymbol{A} \leftarrow \alpha \boldsymbol{x} \boldsymbol{y}^{\mathrm{T}} + \alpha \boldsymbol{y} \boldsymbol{x}^{\mathrm{T}} + \boldsymbol{A}$ | 秩 2 修正 |
| _SYRK | $\boldsymbol{C} \leftarrow \alpha \boldsymbol{A} \boldsymbol{A}^{\mathrm{T}} + \beta \boldsymbol{C}$ | 秩 $k$ 修正 |
| _SYR2K | $\boldsymbol{C} \leftarrow \alpha \boldsymbol{A} \boldsymbol{B}^{\mathrm{T}} + \alpha \boldsymbol{B} \boldsymbol{A}^{\mathrm{T}} + \beta \boldsymbol{C}$ | 秩 $2k$ 修正 |
| _SYMM | $\boldsymbol{C} \leftarrow \alpha \boldsymbol{A} \boldsymbol{B} + \beta \boldsymbol{C}$或$(\alpha \boldsymbol{B} \boldsymbol{A} + \beta \boldsymbol{C})$ | 对称/一般乘积 |

| LAPACK：一些带状/三角运算 | | |
|---|---|---|
| _GBMV | $y \leftarrow \alpha \boldsymbol{A}\boldsymbol{x} + \beta \boldsymbol{y}$ | 一般带状 |
| _SBMV | $y \leftarrow \alpha \boldsymbol{A}\boldsymbol{x} + \beta \boldsymbol{y}$ | 对称带状 |
| _TBMV | $x \leftarrow \alpha \boldsymbol{A}\boldsymbol{x}$ | 三角形 |
| _TPMV | $x \leftarrow \alpha \boldsymbol{A}\boldsymbol{x}$ | 三角形 (按列存) |
| _TRMM | $\boldsymbol{B} \leftarrow \alpha \boldsymbol{A}\boldsymbol{B}$(或$\boldsymbol{B}\boldsymbol{A}$) | 三角形/一般乘积 |

## 1.1 基本算法与记号

矩阵计算是基于线性代数运算的. 点积运算包括标量的加法和乘法. 矩阵向量相乘由点积组成, 矩阵与矩阵相乘相当于一系列的矩阵向量相乘. 所有这些运算都可以用算法形式或者用线性代数的语言来描述. 这一节的主要任务是说明这两种表达方式如何互补. 我们将逐步引入一些记号并且让读者熟悉矩阵计算领域最根本的思维方式. 讨论将围绕着矩阵相乘, 这种计算有几种不同的形式.

### 1.1.1 矩阵记号

令 $\mathbb{R}$ 表示实数集合, $\mathbb{R}^{m\times n}$ 表示所有 $m$ 行 $n$ 列矩阵组成的向量空间:

$$\boldsymbol{A} \in \mathbb{R}^{m\times n} \Leftrightarrow \boldsymbol{A} = (a_{ij}) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R}.$$

如果一个大写字母 (如 $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\Delta}$) 表示一个矩阵, 则带下标 $ij$ 的对应的小写字母 (如 $a_{ij}$, $b_{ij}$, $\delta_{ij}$) 表示矩阵在 $(i,j)$ 的元素. 适当时, 我们也用 $[\boldsymbol{A}]_{ij}$ 和 $\boldsymbol{A}(i,j)$ 表示矩阵元素.

### 1.1.2 矩阵运算

基本矩阵运算包括转置($\mathbb{R}^{m\times n} \to \mathbb{R}^{n\times m}$),

$$\boldsymbol{C} = \boldsymbol{A}^{\mathrm{T}} \Rightarrow c_{ij} = a_{ji};$$

相加($\mathbb{R}^{m\times n} + \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$),

$$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B} \Rightarrow c_{ij} = a_{ij} + b_{ij};$$

标量与矩阵相乘 ($\mathbb{R} \times \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$),

$$\boldsymbol{C} = \alpha \boldsymbol{A} \Rightarrow c_{ij} = \alpha a_{ij};$$

矩阵与矩阵相乘 ($\mathbb{R}^{m\times p} \times \mathbb{R}^{p\times n} \to \mathbb{R}^{m\times n}$),

$$\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B} \Rightarrow c_{ij} = \sum_{k=1}^{r} a_{ik} b_{kj};$$

这些运算是构建矩阵计算的基石.

### 1.1.3 向量记号

令 $\mathbb{R}^n$ 表示 $n$ 维实向量空间:

$$x \in \mathbb{R}^n \Leftrightarrow \boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad x_i \in \mathbb{R}.$$

我们记 $x_i$ 为 $\boldsymbol{x}$ 的第 $i$ 个分量. 根据文中需要, 我们有时也用 $[x]_i$ 和 $x(i)$ 表示 $x_i$.

注意到 $\mathbb{R}^n$ 等于 $\mathbb{R}^{n\times 1}$, 所以 $\mathbb{R}^n$ 中的元素是列向量. 另一方面, $\mathbb{R}^{1\times n}$ 中的元素是行向量:

$$\boldsymbol{x} \in \mathbb{R}^{1\times n} \Leftrightarrow \boldsymbol{x} = (x_1, \cdots, x_n).$$

如果 $\boldsymbol{x}$ 是一个列向量, 则 $\boldsymbol{y} = \boldsymbol{x}^{\mathrm{T}}$ 是行向量.

### 1.1.4 向量运算

假定 $a \in \mathbb{R}$, $\boldsymbol{x} \in \mathbb{R}^n$ 和 $\boldsymbol{y} \in \mathbb{R}^n$. 基本向量运算包括标量与向量相乘,

$$\boldsymbol{z} = a\boldsymbol{x} \Rightarrow z_i = ax_i;$$

向量加法,

$$\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{y} \Rightarrow z_i = x_i + y_i;$$

点积(或称内积):

$$c = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} \Rightarrow c = \sum_{i=1}^{n} x_i y_i;$$

向量相乘(或称 Hadamard积),

$$\boldsymbol{z} = \boldsymbol{x}.*\boldsymbol{y} \Rightarrow z_i = x_i y_i.$$

另一个非常重要的运算是 saxpy, 它具有修正形式:

$$\boldsymbol{y} = a\boldsymbol{x} + \boldsymbol{y} \Rightarrow y_i = ax_i + y_i.$$

这里, 符号“=”用来表示赋值而不是数学上的等号. 此运算修正向量 $\boldsymbol{y}$, 它的名称 saxpy 是在 LAPACK 中所用的. LAPACK 是一个实现了本书中许多算法的软件包. saxpy 是标量 $a$ 乘 $\boldsymbol{x}$ 加 $\boldsymbol{y}$(scalar $a$ $x$ plus $y$) 的缩写.

### 1.1.5 点积和 saxpy 计算

我们用 MATLAB 语言的形式来描述算法, MATLAB 是一种对矩阵计算非常理想的有效的交互系统. 在这一章我们将逐步引入 MATLAB 的记号, 首先我们给出一个计算点积的算法.

**算法 1.1.1(点积)** 如果 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, 此算法计算点积 $c = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$.

$c = 0$

**for** $i = 1:n$

  $c = c + x(i)y(i)$

**end**

两个 $n$ 维向量的点积涉及 $n$ 次乘法和 $n$ 次加法, 所以它是 $O(n)$ 运算, 即工作量是维数的线性函数. saxpy 也是 $O(n)$ 运算, 但它的返回值是一个向量而不是标量.

**算法 1.1.2(saxpy)** 给定 $x, y \in \mathbb{R}^n$ 和 $a \in \mathbb{R}$, 此算法用 $ax+y$ 覆盖 $y$.

```
for i = 1 : n
    y(i) = ax(i) + y(i)
end
```

必须强调的是, 本书所给出的算法是关键的计算思想的框架而不是成品代码.

### 1.1.6 矩阵与向量相乘和 gaxpy

假定 $A \in \mathbb{R}^{m\times n}, x \in \mathbb{R}^n$ 和 $y \in \mathbb{R}^m$, 我们需要计算

$$y = Ax + y.$$

这是广义的 saxpy 运算, 称之为 gaxpy. 此计算的常规方式是每次修正一个分量, 即

$$y_i = \sum_{j=1}^{n} a_{ij}x_j + y_i, \quad i = 1:m.$$

这就是如下的算法:

**算法 1.1.3(gaxpy: 行型)** 设 $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$ 和 $y \in \mathbb{R}^m$, 本算法用 $Ax+y$ 覆盖 $y$.

```
for i = 1 : m
    for  j = 1 : n
        y(i) = A(i, j)x(j) + y(i)
    end
end
```

$Ax$ 可表示为 $A$ 的列向量的线性组合, 如

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 1\times 7 + 2\times 8 \\ 3\times 7 + 4\times 8 \\ 5\times 7 + 6\times 8 \end{bmatrix} = 7\begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} + 8\begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix} = \begin{bmatrix} 23 \\ 53 \\ 83 \end{bmatrix}.$$

这一计算方式导致如下算法.

**算法 1.1.4(gaxpy: 列型)** 设 $A \in \mathbb{R}^{m\times n}, x \in \mathbb{R}^n$ 和 $y \in \mathbb{R}^m$. 本算法用 $Ax+y$ 覆盖 $y$.

```
for j = 1 : n
    for i = 1 : m
        y(i) = A(i, j)x(j) + y(i)
    end
end
```

注意到这两个 gaxpy 算法的内层循环都是 saxpy 运算. 列型的算法是从向量层次重新理解矩阵与向量相乘所导出的, 它也可从行型算法简单地交换循环顺序而得到. 在矩阵计算中, 将循环顺序的交换与对应的线性代数联系起来是重要的.

### 1.1.7 矩阵划分成行和列

算法 1.1.3 和算法 1.1.4 分别按行和按列用 $\boldsymbol{A}$ 的数据. 为了更清楚地说明这一点, 我们引入*矩阵划分*的语言.

从行来看, 一个矩阵是由行向量堆成的:

$$\boldsymbol{A} \in \mathbb{R}^{m\times n} \Leftrightarrow \boldsymbol{A} = \begin{bmatrix} \boldsymbol{r}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{r}_m^{\mathrm{T}} \end{bmatrix}, \quad \boldsymbol{r}_k \in \mathbb{R}^n. \tag{1.1.1}$$

这称之为 $\boldsymbol{A}$ 的*行划分*. 所以, 当我们行划分

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

时, 我们就把 $\boldsymbol{A}$ 看成是由行向量

$$\boldsymbol{r}_1^{\mathrm{T}} = [1 \quad 2], \quad \boldsymbol{r}_2^{\mathrm{T}} = [3 \quad 4], \quad 和 \quad \boldsymbol{r}_3^{\mathrm{T}} = [5 \quad 6]$$

组成的. 利用行划分 (1.1.1), 算法 1.1.3 可表示成

**for** $i = 1 : m$

  $y_i = r_i^{\mathrm{T}} x + y(i)$

**end**

另一方面, 矩阵也是由列向量组成的:

$$\boldsymbol{A} \in \mathbb{R}^{m\times n} \Leftrightarrow \boldsymbol{A} = [\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n], \ \boldsymbol{c}_k \in \mathbb{R}^m. \tag{1.1.2}$$

我们把这称为 $\boldsymbol{A}$ 的*列划分*. 在上面的 $3\times 2$ 的例子中, $\boldsymbol{c}_1$ 和 $\boldsymbol{c}_2$ 就分别是 $\boldsymbol{A}$ 的第一列和第二列:

$$c_1 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}, \quad c_2 = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}.$$

利用 (1.1.2), 我们看到算法 1.1.4 是一个用到 $\boldsymbol{A}$ 的列向量的 saxpy

**for** $j = 1 : n$

  $y = x_j c_j + y$

**end**

在此, $\boldsymbol{y}$ 可看成重复 saxpy 修正的向量连续求和.

### 1.1.8 冒号记号

表示矩阵的一行或一列的一种简便方式是“冒号”记号. 如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, 则 $A(k,:)$ 表示 $\boldsymbol{A}$ 的第 $k$ 行, 即

$$A(k,:) = [a_{k1}, \cdots, a_{kn}].$$

第 $k$ 列表示为

$$A(:,k) = \begin{bmatrix} a_{1k} \\ \vdots \\ a_{mk} \end{bmatrix}.$$

有了这些常规的记号, 我们可将算法 1.1.3 和算法 1.1.4 分别改写成

**for** $i = 1:m$
  $y(i) = A(i,:)x + y(i)$
**end**

和

**for** $j = 1:n$
  $y = x(j)A(:,j) + y$
**end**

利用冒号记号, 我们可省略迭代细节. 因而我们可从向量层次来考虑并且把注意力集中在大的计算问题上.

### 1.1.9 外积修正

作为冒号记号的初步应用, 我们用它来了解外积修正

$$\boldsymbol{A} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}, \quad \boldsymbol{A} \in \mathbb{R}^{m\times n}, \quad \boldsymbol{x} \in \mathbb{R}^{m}, \quad \boldsymbol{y} \in \mathbb{R}^{n}.$$

外积算子 $\boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}$ 由于是两个“细长”矩阵的乘积而“看上去滑稽”. 但它是完全合法的, 因为左边矩阵 $\boldsymbol{x}$ 的列数与右边矩阵 $\boldsymbol{y}^{\mathrm{T}}$ 的行数相等. 例如

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} [4 \quad 5] = \begin{bmatrix} 4 & 5 \\ 8 & 10 \\ 12 & 15 \end{bmatrix}.$$

外积修正的元素可表述为

**for** $i = 1:m$
  **for** $j = 1:n$
    $a_{ij} = a_{ij} + x_i y_j$
  **end**
**end**

对 $j$ 的循环是将 $\boldsymbol{y}^{\mathrm{T}}$ 的倍数加到 $\boldsymbol{A}$ 的第 $i$ 行, 即

**for** $i = 1:m$

$A(i,:) = A(i,:) + x(i)y^{\mathrm{T}}$

**end**

另一方面, 如果我们把对 $i$ 的循环换成内循环, 则是将 $x$ 的倍数加到 $\boldsymbol{A}$ 的第 $j$ 列:

**for** $j = 1:n$

$A(:,j) = A(:,j) + y(j)x$

**end**

注意这两个外积修正算法都是由一组 saxpy 修正所组成的.

### 1.1.10 矩阵与矩阵相乘

考虑 $2\times 2$ 矩阵与矩阵相乘 $\boldsymbol{AB}$. 在点积形式下, 每个元素都可按点积计算:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1\times 5+2\times 7 & 1\times 6+2\times 8 \\ 3\times 5+4\times 7 & 3\times 6+4\times 8 \end{bmatrix}.$$

在 saxpy 形式下, 乘积的每列可看成是 $\boldsymbol{A}$ 之列的线性组合:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \left[5\begin{bmatrix} 1 \\ 3 \end{bmatrix} + 7\begin{bmatrix} 2 \\ 4 \end{bmatrix},\quad 6\begin{bmatrix} 1 \\ 3 \end{bmatrix} + 8\begin{bmatrix} 2 \\ 4 \end{bmatrix}\right].$$

最后, 在外积形式下, 矩阵相乘可看成是一组外积之和:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}[5 \quad 6] + \begin{bmatrix} 2 \\ 4 \end{bmatrix}[7 \quad 8].$$

这几种矩阵相乘的形式虽然在数学上等价, 但由于流量的不同, 在计算上的表现可能是不同的. 这一点将在 1.4 节中进一步讨论. 现在, 我们详细给出矩阵相乘的上述 3 种形式, 这将有助于我们加深对冒号记号概念的理解以及习惯从不同线性代数的层次上来思考.

### 1.1.11 标量描述

为讨论集中, 我们考虑如下矩阵相乘的修正公式:

$$\boldsymbol{C} = \boldsymbol{AB} + \boldsymbol{C},\quad \boldsymbol{A}\in\mathbb{R}^{m\times p},\quad \boldsymbol{B}\in\mathbb{R}^{p\times n},\quad \boldsymbol{C}\in\mathbb{R}^{m\times n}.$$

首先是熟悉的三层循环算法:

**算法 1.1.5 (矩阵乘法: $ijk$ 形式)** 给出 $\boldsymbol{A}\in\mathbb{R}^{m\times p}, \boldsymbol{B}\in\mathbb{R}^{p\times n}$ 和 $\boldsymbol{C}\in\mathbb{R}^{m\times n}$, 本算法用 $\boldsymbol{AB}+\boldsymbol{C}$ 覆盖 $\boldsymbol{C}$.

**for** $i = 1:m$

  **for** $j = 1:n$

    **for** $k = 1:p$

      $C(i,j) = A(i,k)B(k,j) + C(i,j)$

    **end**

```
        end
    end
```

由于我们把 $C$(以及 $A$) 的行标为 $i$, $C$(以及 $B$) 的列标为 $j$, 求和的下标记为 $k$, 所以这是一个 $ijk$ 形式.

我们考虑修正公式 $C = AB + C$ 而不是 $C = AB$ 有两个理由: 一是不必费心考虑 $C = 0$ 的初始化; 二是在实际中 $C = AB + C$ 出现得更频繁.

矩阵相乘的修正中 3 个求和循环可任意排序, 从而一共有 3!=6 种形式. 所以,

```
for j = 1 : n
    for k = 1 : p
        for i = 1 : m
            C(i, j) = A(i, k)B(k, j) + C(i, j)
        end
    end
end
```

是 $jki$ 形式. 这 6 种可能性 $(ijk, jik, ikj, jki, kij, kji)$ 的任何一个都对应一内层循环运算 (点积或 saxpy), 而且具有自己的数据流动模式. 例如, 在 $ijk$ 形式, 内层循环是点积, 数据用到的是 $A$ 的行和 $B$ 的列. 而在 $jki$ 形式下内层循环是 saxpy, 数据用到的是 $C$ 的列和 $A$ 的列. 这些性质以及把中层循环与内层循环连在一起考虑是代表何种运算都归纳在表 1.1.1 中. 每种形式的浮点运算次数都一样, 但对 $A, B, C$ 数据的存取却是不同的.

**表 1.1.1 矩阵乘法: 循环排序及特性**

| 循环顺序 | 内层循环 | 中层循环 | 内层数据存取 |
|---|---|---|---|
| $ijk$ | 点积 | 向量乘矩阵 | $A$ 的行, $B$ 的列 |
| $jik$ | 点积 | 矩阵乘向量 | $A$ 的行, $B$ 的列 |
| $ikj$ | saxpy | 行的 gaxpy | $B$ 的行, $C$ 的行 |
| $jki$ | saxpy | 列的 gaxpy | $A$ 的列, $C$ 的列 |
| $kij$ | saxpy | 行的外积 | $B$ 的行, $C$ 的行 |
| $kji$ | saxpy | 列的外积 | $A$ 的列, $C$ 的列 |

### 1.1.12 点积形式

通常的矩阵乘法过程把 $AB$ 视为用点积计算的数组, 该数组可按从左到右, 从上到下的次序逐个计算. 这就是算法 1.1.5 所用到的思想. 利用冒号记号可突显其点积形式.

**算法 1.1.6 (矩阵乘法: 点积形式)** 给出 $A \in \mathbb{R}^{m\times p}, B \in \mathbb{R}^{p\times n}$ 和 $C \in \mathbb{R}^{m\times n}$, 本算法用 $AB + C$ 覆盖 $C$.

```
for i = 1 : m
```

for $j = 1 : n$

$C(i, j) = A(i, :)B(:, j) + C(i, j)$

end

end

利用矩阵划分语言, 记

$$A = \begin{bmatrix} a_1^{\mathrm{T}} \\ \vdots \\ a_m^{\mathrm{T}} \end{bmatrix}, \quad a_k \in \mathbb{R}^p,$$

$$B = [b_1, \cdots, b_n], \quad b_k \in \mathbb{R}^p,$$

则算法 1.1.6 可表示为

for $i = 1 : m$

for $j = 1 : n$

$c_{ij} = a_i^{\mathrm{T}} b_j + c_{ij}$

end

end

注意到 $j$ 循环的“任务”是计算修正的第 $i$ 行. 为强调这一点, 可写成

for $i = 1 : m$

$c_i^{\mathrm{T}} = a_i^{\mathrm{T}} B + c_i^{\mathrm{T}}$

end

这里

$$C = \begin{bmatrix} c_1^{\mathrm{T}} \\ \vdots \\ c_m^{\mathrm{T}} \end{bmatrix}$$

是 $C$ 的行划分. 用冒号记号来表示这同一运算, 可写成

for $i = 1 : m$

$C(i, :) = A(i, :)B + C(i, :)$

end

无论从哪种形式我们都看到, $ijk$ 形式的内部的两层循环定义一个基于行的 gaxpy 运算.

### 1.1.13 saxpy 形式

假定 $A$ 和 $C$ 列划分为

$$A = [a_1, \cdots, a_p], \quad a_j \in \mathbb{R}^m,$$
$$C = [c_1, \cdots, c_n], \quad c_j \in \mathbb{R}^m.$$

比较 $C = AB + C$ 两边的第 $j$ 列可知

$$c_j = \sum_{k=1}^{p} b_{kj} a_k + c_j, \quad j = 1:n.$$

这些向量求和可通过一系列 saxpy 修正 (运算) 集中起来.

**算法 1.1.7 (矩阵乘法: saxpy 形式)** 给出 $A \in \mathbb{R}^{m \times p}$, $B \in \mathbb{R}^{p \times n}$, $C \in \mathbb{R}^{m \times n}$, 本算法用 $AB + C$ 覆盖 $C$.

```
for j = 1 : n
    for k = 1 : p
        C(:, j) = A(:, k)B(k, j) + C(:, j)
    end
end
```

注意到 $k$ 循环实质上是 gaxpy 运算:

```
for j = 1 : n
    C(:, j) = AB(:, j) + C(:, j)
end
```

### 1.1.14 外积形式

考虑算法 1.1.5 的 $kji$ 形式:

```
for k = 1 : p
    for j = 1 : n
        for i = 1 : m
            C(i, j) = A(i, k)B(k, j) + C(i, j)
        end
    end
end
```

内部的两层循环是一个外积修正

$$C = a_k b_k^{\mathrm{T}} + C$$

其中 $a_k \in \mathbb{R}^m$, $b_k \in \mathbb{R}^n$ 且

$$A = [a_1, \cdots, a_p] \quad 和 \quad B = \begin{bmatrix} b_1^{\mathrm{T}} \\ \vdots \\ b_p^{\mathrm{T}} \end{bmatrix}. \tag{1.1.3}$$

因而, 我们得到如下算法.

**算法 1.1.8 (矩阵乘法: 外积形式)** 给出 $A \in \mathbb{R}^{m \times p}$, $B \in \mathbb{R}^{p \times n}$ 和 $C \in \mathbb{R}^{m \times n}$, 本算法用 $AB + C$ 覆盖 $C$.

```
for k = 1 : p
    C = A(:, k)B(k, :) + C
```

**end**

这一实现方法是基于 $\boldsymbol{AB}$ 是 $p$ 个外积之和.

### 1.1.15 “级”的概念

点积和 saxpy 运算是“1 级”运算的例子. 1 级运算涉及的数据量和运算量都是运算维数的线性函数. $m\times n$ 的外积修正和 gaxpy 运算涉及二次数据量 $(O(mn))$ 和二次运算量 $(O(mn))$. 它们是“2 级”运算的例子.

矩阵修正公式 $\boldsymbol{C}=\boldsymbol{AB}+\boldsymbol{C}$ 是“3 级”运算. 3 级运算涉及二次数据量和三次运算量, 设 $\boldsymbol{A},\boldsymbol{B}$ 和 $\boldsymbol{C}$ 都是 $n\times n$ 矩阵, 则 $\boldsymbol{C}=\boldsymbol{AB}+\boldsymbol{C}$ 需要 $O(n^2)$ 内存和 $O(n^3)$ 运算量.

本书中一个反复出现的主题是设计具有高“级”线性代数运算的矩阵算法. 例如, 一个高性能的线性方程组的算法可能需要把高斯消元法在 3 级上来组织. 要做到这一点就必须在运算上重新考虑, 因为方法通常是用 1 级的方式给出的, 例如“把第一行乘以一个标量后加到第二行”.

### 1.1.16 矩阵方程

为更好地通过外积来理解矩阵乘法, 我们实际上建立了矩阵方程

$$\boldsymbol{AB}=\sum_{k=1}^{p}\boldsymbol{a}_k\boldsymbol{b}_k^{\mathrm{T}}$$

其中 $\boldsymbol{a}_k$ 和 $\boldsymbol{b}_k$ 是由 (1.1.3) 中划分所定义的.

在以后的章节中, 大量的矩阵方程会出现. 有时它们会像上面外积展开一样以算法形式建立, 有时它们是在 $ij$ 元素这个层次上来证明的. 作为后者的一个例子, 我们证明乘积转置的性质这一重要结果.

**定理 1.1.1** *设* $\boldsymbol{A}\in\mathbb{R}^{m\times p}$ *和* $\boldsymbol{B}\in\mathbb{R}^{p\times n}$, *则* $(\boldsymbol{AB})^{\mathrm{T}}=\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}$.

**证明** 令 $\boldsymbol{C}=(\boldsymbol{AB})^{\mathrm{T}}$, 则

$$c_{ij}=[(\boldsymbol{AB})^{\mathrm{T}}]_{ij}=[\boldsymbol{AB}]_{ji}=\sum_{k=1}^{p}a_{jk}b_{ki}.$$

另一方面, 设 $\boldsymbol{D}=\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}$, 则

$$d_{ij}=[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}]_{ij}=\sum_{k=1}^{p}[\boldsymbol{B}^{\mathrm{T}}]_{ik}[\boldsymbol{A}^{\mathrm{T}}]_{kj}=\sum_{k=1}^{p}b_{ki}a_{jk}.$$

由于对所有 $i$ 和 $j$ 都有 $c_{ij}=d_{ij}$, 故知 $\boldsymbol{C}=\boldsymbol{D}$. □

像上面这样的基于标量的证明常常没有启发性, 但有时却是唯一的方式.

### 1.1.17 复矩阵

有时我们要讨论复矩阵的计算. $m\times n$ 复矩阵的向量空间记为 $\mathbb{C}^{m\times n}$. 复矩阵的标量乘、相加、相乘是与实矩阵完全相对应的. 但是, 转置在复情形下是**转置共**

轭:

$$C = A^{\mathrm{H}} \Rightarrow c_{ij} = \bar{a}_{ji}.$$

$n$ 维复向量的向量空间记为 $\mathbb{C}^n$. $n$ 维复向量 $x$ 和 $y$ 的点积是

$$s = x^{\mathrm{H}} y = \sum_{i=1}^{n} \bar{x}_i y_i.$$

最后, 假定 $A = B + \mathrm{i}C \in \mathbb{C}^{m\times n}$, 则我们把 $A$ 的实部与虚部分别记为 $\mathrm{Re}(A) = B$ 和 $\mathrm{Im}(A) = C$.

## 习 题

**1.1.1** 假定 $A \in \mathbb{R}^{n\times n}$ 和 $x \in \mathbb{R}^r$, 给出一个计算 $M = (A - x_1 I)\cdots(A - x_r I)$ 的第一列的 saxpy 算法.

**1.1.2** 在常规的 2×2 矩阵相乘 $C = AB$ 中有 8 个乘法:

$$a_{11}b_{11}, a_{11}b_{12}, a_{21}b_{11}, a_{21}b_{12}, a_{12}b_{21}, a_{12}b_{22}, a_{22}b_{21}, a_{22}b_{22}.$$

对 $ijk$, $jik$, $kij$, $ikj$, $jki$ 及 $kji$ 矩阵乘法形式, 列表给出这些乘法的顺序.

**1.1.3** 给出一个计算 $C = (xy^{\mathrm{T}})^k$ 的算法, 其中 $x$ 和 $y$ 是 $n$ 维向量.

**1.1.4** 描述一个求 $(XY^T)^k$ 的算法, 其中 $X, Y \in \mathbb{R}^{n\times 2}$.

**1.1.5** 给出一个外积形式的算法修正 $C = AB^{\mathrm{T}} + C$, 其中 $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times r}$, $C \in \mathbb{R}^{m\times n}$.

**1.1.6** 假定 $C, D, E$ 和 $F$ 是给定的 $n \times n$ 实矩阵. 说明如何只计算三次实的 $n \times n$ 矩阵乘法就可求出实的 $n \times n$ 矩阵 $A$ 和 $B$, 使得 $(A + \mathrm{i}B) = (C + \mathrm{i}D)(E + \mathrm{i}F)$. 提示: 计算 $W = (C + D)(E - F)$.

## 本节注释与参考文献

必须强调的是, 从我们的"半正式"的算法描述到高水平软件的开发仍需很大的努力. 就连 1 级、2 级和 3 级的 BLAS 的实现也要做精细的工作.

C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh(1979). "Basic Linear Algebra Subprograms for FORTRAN Usage," *ACM Trans. Math. Soft.* 5, 308–323.

C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh(1979). "Algorithm 539, Basic Linear Algebra Subprograms for FORTRAN Usage," *ACM Trans. Math. Soft.* 5, 324–325.

J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson(1988). "An Extended Set of Fortran Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.* 14, 1–17.

J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson(1988). "Algorithm 656 An Extended Set of Fortran Basic Linear Algebra Subprograms: Model Implementation and Test Programs," *ACM Trans. Math. Soft.* 14, 18–32.

J. J. Dongarra, J. Du Croz, I. S. Duff, and S. J. Hammarling(1990). "A Set of Level 3 Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.* 16, 1–17.

J. J. Dongarra, J. Du Croz, I. S. Duff, and S. J. Hammarling(1990). "Algorithm 679. A Set of Level 3 Basic Linear Algebra Subprograms: Model Implementation and Test Programs," *ACM Trans. Math. Soft.* 16, 18–28.

其他关于 BLAS 的文献包括如下.

B. Kågström, P. Ling, and C. Van Loan(1991). "High-Performance Level-3 BLAS: Sample Routines for Double Precision Real Data," in *High Performance Computing II*, M. Durand and F. El Dabaghi(eds), North-Holland. 269–281.

B. Kågström, P. Ling and C. Van Loan(1995). "GEMM-Based Level-3 BLAS: High-Performance Model Implementations and Performance Evaluation Benchmark," in *Parallel Programming and Applications*, P. Fritzon and L. Finmo(eds), ISO Press, 184–188.

关于软件设计技巧的论述我们推荐:

J. R. Rice(1981). *Matrix Computations and Mathematical Software*, Academic Press, New York.

还请参阅 LAPACK 手册.

## 1.2 利用结构

一个给定的矩阵算法的效率是和许多东西相关的. 最明显的, 也是本节我们要考虑的, 是运算量和存储量. 我们继续用矩阵与向量相乘以及矩阵与矩阵相乘为工具来介绍主要的思想. 作为可利用的结构的例子, 我们挑选了带状矩阵和对称矩阵. 带状矩阵有许多零元素, 因而在带状矩阵计算中可省去许多运算量和存储量, 这是不足为奇的. 下面将讨论此类矩阵计算的运算复杂度和数据结构.

对称矩阵提供了利用结构的另一类例子. 对称线性方程组以及对称特征值问题在矩阵计算中起显著作用, 所以熟悉它们的技巧是重要的.

### 1.2.1 带状矩阵和 x-0 记号

如果对任何 $i > j + p$ 都有 $a_{ij} = 0$, 就称 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 具有下带宽$p$; 如果对任何 $j > i + q$ 都有 $a_{ij} = 0$, 则称 $\boldsymbol{A}$ 具有上带宽 $q$. 下面是一个具有下带宽 1 和上带宽 2 的 $8 \times 5$ 矩阵:

$$\begin{bmatrix} \times & \times & \times & 0 & 0 \\ \times & \times & \times & \times & 0 \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

"×"表示任意非零元素. 这个记号表示矩阵的非零结构很方便, 我们会大量使用, 常见的带状矩阵在表 1.2.1 中列出.

表 1.2.1　$m\times n$ 带状矩阵术语

| 矩阵类型 | 下带宽 | 上带宽 |
|---|---|---|
| 对角矩阵 | 0 | 0 |
| 上三角形矩阵 | 0 | $n-1$ |
| 下三角形矩阵 | $m-1$ | 0 |
| 三对角矩阵 | 1 | 1 |
| 上双对角矩阵 | 0 | 1 |
| 下双对角矩阵 | 1 | 0 |
| 上 Hessenberg 矩阵 | 1 | $n-1$ |
| 下 Hessenberg 矩阵 | $m-1$ | 1 |

### 1.2.2　对角矩阵

上下带宽都为零的矩阵是对角矩阵. 设 $\boldsymbol{D}\in\mathbb{R}^{m\times n}$ 是对角矩阵, 则

$$\boldsymbol{D}=\mathrm{diag}(d_1,\cdots,d_q),\quad q=\min\{m,n\}\Leftrightarrow d_i=d_{ii}.$$

设 $\boldsymbol{D}$ 是对角矩阵, $\boldsymbol{A}$ 是一般矩阵, 则 $\boldsymbol{DA}$ 是 $\boldsymbol{A}$ 的行加权, $\boldsymbol{AD}$ 是 $\boldsymbol{A}$ 的列加权.

### 1.2.3　三角形矩阵乘法

为引入带状矩阵"思维", 我们考虑矩阵相乘问题 $\boldsymbol{C}=\boldsymbol{AB}$, 其中 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是 $n\times n$ 上三角形矩阵. $3\times 3$ 的情形可表示为

$$\boldsymbol{C}=\begin{bmatrix} a_{11}b_{11} & a_{11}b_{12}+a_{12}b_{22} & a_{11}b_{13}+a_{12}b_{23}+a_{13}b_{33} \\ 0 & a_{22}b_{22} & a_{22}b_{23}+a_{23}b_{33} \\ 0 & 0 & a_{33}b_{33} \end{bmatrix}.$$

从上可知, 乘积是上三角形矩阵, 而且在上三角的元素是简化了的内积. 事实上, 对任何 $k<i$ 或 $j<k$ 都有 $a_{ik}b_{kj}=0$, 我们可看到

$$c_{ij}=\sum_{k=i}^{j}a_{ik}b_{kj},$$

于是得到下列算法.

**算法 1.2.1 (三角形矩阵乘法)**　给出上三角形矩阵 $\boldsymbol{A},\boldsymbol{B}\in\mathbb{R}^{n\times n}$, 本算法计算 $\boldsymbol{C}=\boldsymbol{AB}$.

```
C = 0
for i = 1 : n
    for j = i : n
        for k = i : j
```

$$C(i,j) = A(i,k)B(k,j) + C(i,j)$$

**end**

**end**

**end**

为量化这个算法节省的计算量, 我们需要一些度量计算量的工具.

### 1.2.4 flop

很显然, 比起满矩阵相乘, 上三角形矩阵相乘所需要的计算量要小. 量化这一点的方式之一是通过 flop 记号. 一个 flop① 就是一个浮点运算. 长度为 $n$ 的内积或 saxpy 运算需要 $2n$ 个 flop, 因为它们都需要 $n$ 次乘法和 $n$ 次加法.

对于 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, gaxpy 运算 $\boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{y}$ 和 $m \times n$ 外积修正 $\boldsymbol{A} = \boldsymbol{A} + \boldsymbol{xy}^{\mathrm{T}}$ 都需要 $2mn$ 个 flop.

对于 $\boldsymbol{A} \in \mathbb{R}^{m\times p}$, $\boldsymbol{B} \in \mathbb{R}^{p\times n}$ 和 $\boldsymbol{C} \in \mathbb{R}^{m\times n}$, 矩阵乘法修正 $\boldsymbol{C} = \boldsymbol{AB} + \boldsymbol{C}$ 需要 $2mnp$ 个 flop.

flop 数通常可通过将算法最内层循环的运算数相加而得到, 以矩阵乘法为例, 最内层语句是

$$C(i,j) = A(i,k)B(k,j) + C(i,j),$$

它有 2 个 flop. 通过简单的循环计数知, 此语句要执行 $mnp$ 次, 所以一般矩阵相乘需要 $2mnp$ 个 flop.

现在, 我们研究算法 1.2.1 的运算量. 注意到 $c_{ij}(i \leqslant j)$ 需要 $2(j-i+1)$ 个 flop. 利用直观公式

$$\sum_{p=1}^{q} p = \frac{q(q+1)}{2} \approx \frac{q^2}{2},$$

$$\sum_{p=1}^{q} p^2 = \frac{q^3}{3} + \frac{q^2}{2} + \frac{q}{6} \approx \frac{q^3}{3},$$

我们发现三角形矩阵乘法约需满矩阵乘法的 flop 数的六分之一:

$$\sum_{i=1}^{n}\sum_{j=i}^{n} 2(j-i+1) = \sum_{i=1}^{n}\sum_{j=1}^{n-i+1} 2j \approx \sum_{i=1}^{n} \frac{2(n-i+1)^2}{2}$$

$$= \sum_{i=1}^{n} i^2 \approx \frac{n^3}{3}.$$

我们舍去了低阶项, 因为它们对 flop 数影响甚微. 例如, 通过精确的浮点数统计可发现算法 1.2.1 需 $n^3/3 + n^2 + 2n/3$ 个 flop. 当 $n$ 很大时 (这通常是我们感兴趣的情形), 我们看到, 精确的 flop 数与 $n^3/3$ 近似估计相比, 多出的两项没有实质的意义.

① 在本书第 1 版中, 我们定义一个 flop 是与形如 $a_{ij} = a_{ij} + a_{ik}a_{kj}$ 的运算相关的工作量, 即一个浮点加、一个浮点乘和下标. 因此"旧 flop"包含两个"新 flop". 把 flop 定义为单个浮点运算, 我们能得到算法复杂度的更精确的度量.

用 flop 数来衡量程序的效率必然是粗糙的, 因为它忽略了下标和流量, 也没考虑其他执行程序的负荷. 我们不能过分依赖于 flop 数的比较. 例如, 我们不能断定三角形矩阵相乘要比方阵相乘快 6 倍. flop 数只是一个"快而不正当"的评价方法, 它仅注意到影响效率的多个因素之一.

### 1.2.5 再论冒号记号

如果我们把 1.1.8 节引入的冒号记号推广, 则算法 1.2.1 中 $k$ 循环的点积可简洁地叙述. 假设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 整数 $p,q$ 和 $r$ 满足 $1\leqslant p\leqslant q\leqslant n$ 和 $1\leqslant r\leqslant m$. 我们定义

$$A(r,p:q)=[a_{rp},\cdots,a_{rq}]\in\mathbb{R}^{1\times(q-p+1)}.$$

同样, 如果 $1\leqslant p\leqslant q\leqslant m$ 和 $1\leqslant c\leqslant n$, 则

$$A(p:q,c)=\begin{bmatrix}a_{pc}\\ \vdots\\ a_{qc}\end{bmatrix}\in\mathbb{R}^{q-p+1}.$$

利用这个记号, 算法 1.2.1 可写成

$C(1:n,1:n)=0$

**for** $i=1:n$

  **for** $j=i:n$

    $C(i,j)=A(i,i:j)B(i:j,j)+C(i,j)$

  **end**

**end**

我们给出冒号记号的另一个性质, 就是允许下标负增长. 所以, 如果 $x$ 和 $y$ 是 $n$ 维向量, 则 $s=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}(n:-1:1)$ 代表求和

$$s=\sum_{i=1}^{n}x_iy_{n-i+1}.$$

### 1.2.6 带状矩阵存储

假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 具有下带宽 $p$ 和上带宽 $q$ 且 $p$ 和 $q$ 远小于 $n$. 这样的矩阵可按下面方式储存在 $(p+q+1)\times n$ 数组 A.band, 即对所有位于带宽之内的 $(i,j)$:

$$a_{ij}=\text{A.band}(i-j+q+1,j). \tag{1.2.1}$$

所以, 如果

$$\boldsymbol{A}=\begin{bmatrix}a_{11}&a_{12}&a_{13}&0&0&0\\ a_{21}&a_{22}&a_{23}&a_{24}&0&0\\ 0&a_{32}&a_{33}&a_{34}&a_{35}&0\\ 0&0&a_{43}&a_{44}&a_{45}&a_{46}\\ 0&0&0&a_{54}&a_{55}&a_{56}\\ 0&0&0&0&a_{65}&a_{66}\end{bmatrix},$$

则

$$\text{A.band} = \begin{bmatrix} 0 & 0 & a_{13} & a_{24} & a_{35} & a_{46} \\ 0 & a_{12} & a_{23} & a_{34} & a_{45} & a_{56} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} \\ a_{21} & a_{32} & a_{43} & a_{54} & a_{65} & 0 \end{bmatrix}.$$

这里, "0" 元素是用不着的, 利用这种数据结构, 我们基于列的 gaxpy 算法可化成如下形式.

**算法 1.2.2 (带矩阵 gaxpy)** 设 $A \in \mathbb{R}^{n\times n}$ 具有下带宽 $p$ 和上带宽 $q$ 且储存在 A.band(1.2.1 节), 如果 $x, y \in \mathbb{R}^n$, 则本算法用 $Ax + y$ 覆盖 $y$.

**for** $j = 1 : n$

$\qquad y_{\text{top}} = \max(1, j - q)$

$\qquad y_{\text{bot}} = \min(n, j + p)$

$\qquad a_{\text{top}} = \max(1, q + 2 - j)$

$\qquad a_{\text{bot}} = a_{\text{top}} + y_{\text{bot}} - y_{\text{top}}$

$\qquad y(y_{\text{top}} : y_{\text{bot}}) = x(j)\text{A.band}(a_{\text{top}} : a_{\text{bot}}, j) + y(y_{\text{top}} : y_{\text{bot}})$

**end**

注意到把 $A$ 按列储存于 A.band, 我们得到一个基于列的 saxpy 程序. 事实上, 算法 1.2.2 是从算法 1.1.4 中把每个 saxpy 涉及的向量换成非零的短向量而得到的. 整数计算用于确定非零元素的位置. 由于细致地进行了零元素及非零元素分析, 此算法在 $p$ 和 $q$ 远小于 $n$ 的假定下仅需 $2n(p+q+1)$ 个 flop.

### 1.2.7 对称性

如果 $A = A^{\mathrm{T}}$, 我们说 $A \in \mathbb{R}^{n\times n}$ 是对称的. 所以,

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$$

是对称的. 如果我们仅储存下三角的元素, 则存储量可减少一半, 如 A.vec=[1 2 3 4 5 6]. 在一般情况下, 根据此约定, 我们把 $a_{ij}$ 按如下方式储存

$$a_{ij} = \text{A.vec}((j-1)n - j(j-1)/2 + i) \quad (i \geqslant j). \tag{1.2.2}$$

下面看看矩阵 $A$ 储存在 A.vec 的基于列的 gaxpy 运算.

**算法 1.2.3 (对称储存 gaxpy)** 设 $A \in \mathbb{R}^{n\times n}$ 对称, 储存于 A. vec(见 (1.2.2)), $x, y \in \mathbb{R}^n$, 本算法用 $Ax + y$ 覆盖 $y$.

**for** $j = 1 : n$

$\qquad$ **for** $i = 1 : j - 1$

$\qquad\qquad y(i) = \text{A.vec}((i-1)n - i(i-1)/2 + j)x(j) + y(i)$

```
        end
        for i = j : n
            y(i) = A.vec((j − 1)n − j(j − 1)/2 + i)x(j) + y(i)
        end
    end
```

此算法与常规的 gaxpy 一样需要 $2n^2$ 个 flop. 值得注意的是减少一半存储换来了很别扭的下标.

### 1.2.8 按对角储存

对称矩阵也可按对角储存. 对于

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix},$$

按对角储存则把 $A$ 存于向量

$$\text{A.diag} = [1 \quad 4 \quad 6 \quad 2 \quad 5 \quad 3].$$

在一般情形, 对 $i \geqslant j$, 有

$$a_{i+k,i} = \text{A.diag}(i + nk - k(k-1)/2) \quad (k \geqslant 0). \tag{1.2.3}$$

为简化在矩阵乘向量中如何运用这一数据结构的讨论, 我们需引入如下记号.

设 $A \in \mathbb{R}^{m\times n}$, 用 $D(A, k) \in \mathbb{R}^{m\times n}$ 表示 $A$ 的第 $k$ 条对角线:

$$[D(A,k)]_{ij} = \begin{cases} a_{ij}, & j = i + k, \quad 1 \leqslant i \leqslant m, \quad 1 \leqslant j \leqslant n, \\ 0, & \text{否则}. \end{cases}$$

这样

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{D(A,2)} + \underbrace{\begin{bmatrix} 0 & 2 & 0 \\ 0 & 0 & 5 \\ 0 & 0 & 0 \end{bmatrix}}_{D(A,1)}$$

$$+ \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 6 \end{bmatrix}}_{D(A,0)} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 5 & 0 \end{bmatrix}}_{D(A,-1)} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 3 & 0 & 0 \end{bmatrix}}_{D(A,-2)}.$$

回到我们按对角储存的数据结构, 我们看到 $D(A,0), D(A,1), \cdots, D(A,n-1)$ 的非零部分是依次存于 A.diag(1.2.3) 里. 于是 gaxpy 运算 $y = Ax + y$ 可组织成

$$y = D(A,0)x + \sum_{k=1}^{n-1}(D(A,k) + D(A,k)^{\mathrm{T}})x + y.$$

将上述公式的细节整理即得到如下算法.

**算法 1.2.4 (按对角储存的 gaxpy)** 设 $A \in \mathbb{R}^{n\times n}$ 对称, 储存于 A.diag(1.2.3) 里 $x, y \in \mathbb{R}^n$, 本算法用 $Ax + y$ 覆盖 $y$.

```
for  i = 1 : n
        y(i) = A.diag(i)x(i) + y(i)
end
for  k = 1 : n − 1
        t = nk − k(k − 1)/2
        {y = D(A, k)x + y}
        for  i = 1 : n − k
                y(i) = A.diag(i + t)x(i + k) + y(i)
        end
        {y = D(A, k)^T x + y}
        for  i = 1 : n − k
                y(i + k) = A.diag(i + t)x(i) + y(i + k)
        end
end
```

注意在内层循环是向量相乘:

$$y(1:n-k) = \text{A.diag}(t+1:t+n-k).*x(k+1:n) + y(1:n-k)$$

$$y(k+1:n) = \text{A.diag}(t+1:t+n-k).*x(1:n-k) + y(k+1:n).$$

### 1.2.9 覆盖和工作空间

上述讨论未涉及存储的经济使用. 覆盖输入数据是控制矩阵计算所需内存量的另一方法. 考虑 $n\times n$ 矩阵相乘问题 $C = AB$, 假定"输入矩阵" $B$ 将被"输出矩阵" $C$ 所覆盖. 我们不能简单地将

```
C(1 : n, 1 : n) = 0
for j = 1 : n
      for  k = 1 : n
              C(:, j) = C(:, j) + A(:, k)B(k, j)
      end
end
```

改成

```
for j = 1 : n
      for  k = 1 : n
              B(:, j) = B(:, j) + A(:, k)B(k, j)
      end
```

```
    end
```

这是因为 $B(:,j)$ 在整个 $k$ 循环中都要用到. 需要一个线性的"工作空间"来暂存乘积的第 $j$ 列直到可以"安全"地覆盖 $B(:,j)$:

```
for j = 1 : n
    w(1 : n) = 0
    for k = 1 : n
        w(:) = w(:) + A(:, k)B(k, j)
    end
    B(:, j) = w(:)
end
```

增加一个线性工作空间在同阶的 2 维数组的矩阵计算中常常是无足轻重的.

## 习 题

**1.2.1** 给出一个用 $\boldsymbol{A}^2$ 覆盖 $\boldsymbol{A}$ 的算法, 其中 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是 (a) 上三角形矩阵,(b) 方阵. 在两种情形下都尽可能少地用工作空间.

**1.2.2** 假定 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是上 Hessenberg 矩阵以及 $\lambda, \cdots, \lambda_r$ 是给定的标量. 请给出计算 $M = (\boldsymbol{A} - \lambda_1 \boldsymbol{I}) \cdots (\boldsymbol{A} - \lambda_r \boldsymbol{I})$ 第一列的 saxpy 算法.

**1.2.3** 给出一个列 saxpy 算法计算 $n \times n$ 矩阵乘积 $\boldsymbol{C} = \boldsymbol{AB}$, 其中 $\boldsymbol{A}$ 是上三角形矩阵, $\boldsymbol{B}$ 是下三角形矩阵.

**1.2.4** 把算法 1.2.2 推广到长方形的带状矩阵. 注意对数据结构的描述.

**1.2.5** 如果 $\boldsymbol{A}^{\mathrm{H}} = \boldsymbol{A}$, 则称 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ 是 Hermite 矩阵. 设 $\boldsymbol{A} = \boldsymbol{B} + \mathrm{i}\boldsymbol{C}$, 则知 $\boldsymbol{B}^{\mathrm{T}} = \boldsymbol{B}$, $\boldsymbol{C}^{\mathrm{T}} = -\boldsymbol{C}$. 假定我们用 A.herm$(i,j)$ 储存 $b_{ij}$(如果 $i \geqslant j$) 和 $c_{ij}$(如果 $j > i$) 的方式把 $\boldsymbol{A}$ 表示在数组 A.herm 中. 利用这一数据结构写出矩阵乘向量的算法从 $\mathrm{Re}(x)$ 和 $\mathrm{Im}(x)$ 计算 $\mathrm{Re}(z)$ 和 Im(z) 使得 $z = \boldsymbol{A}x$.

**1.2.6** 设 $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. $\boldsymbol{A}$ 对称且按对角方式储存. 给出计算 $\boldsymbol{Y} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX}$ 并将结果仍按对角储存的算法. 用不同的数组储存 $\boldsymbol{A}$ 和 $\boldsymbol{Y}$.

**1.2.7** 设 $\boldsymbol{a} \in \mathbb{R}^n$ 给定且 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 满足 $a_{ij} = a_{|i-j|+1}$. 给出一个算法, 用 $\boldsymbol{Ax} + \boldsymbol{y}$ 覆盖 $\boldsymbol{y}$, 其中 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 是任给向量.

**1.2.8** 设 $\boldsymbol{a} \in \mathbb{R}^n$ 给定且 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 满足 $a_{ij} = a_{((i+j-1)_{\bmod n})+1}$, 给出一个算法, 用 $\boldsymbol{Ax} + \boldsymbol{y}$ 覆盖 $\boldsymbol{y}$, 其中 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 是任给向量.

**1.2.9** 求一个紧凑的将非对称带状矩阵按对角储存的方法, 并给出所对应的gaxpy算法.

**1.2.10** 设 $\boldsymbol{p}$ 和 $\boldsymbol{q}$ 是 $n$ 维向量, $\boldsymbol{A} = (a_{ij})$ 由 $a_{ij} = a_{ji} = p_i q_j (1 \leqslant i \leqslant j \leqslant n)$ 所定义. 对给定的 $\boldsymbol{x} \in \mathbb{R}^n$, 计算 $\boldsymbol{y} = \boldsymbol{Ax}$ 需要多少个 flop?

## 本节注释与参考文献

关于对称以及带状数据结构的讨论可参阅 LAPACK 手册. 也可参考:

N. Madsen, G. Roderigue, and J. Karush(1976). "Matrix Multiplication by Diagonals on a Vector Parallel Processor," *Information Processing Letters* 5, 41–45.

## 1.3 分块矩阵和算法

采用分块矩阵记号这一工具在矩阵计算中是很重要的, 因为它可简化许多核心算法的导出. 而且, "分块算法" 在高性能计算中愈来愈重要. 分块算法实质上是指大量用到矩阵乘矩阵的算法. 在许多计算环境下, 这类算法比基于低层次线性代数的算法要有效得多.

### 1.3.1 分块矩阵记号

行划分和列划分是矩阵分块的特殊情形. 在一般情况下, 我们对 $m\times n$ 矩阵 $\boldsymbol{A}$ 的行和列进行划分, 得到

$$\boldsymbol{A}=\begin{array}{c}\left[\begin{array}{ccc}\boldsymbol{A}_{11} & \cdots & \boldsymbol{A}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{A}_{q1} & \cdots & \boldsymbol{A}_{qr}\end{array}\right]\begin{array}{c}m_1\\ \\ m_q\end{array}\\ \begin{array}{ccc}n_1 & \quad & n_r\end{array}\end{array},$$

其中 $m_1+\cdots+m_q=m, n_1+\cdots+n_r=n$, $\boldsymbol{A}_{\alpha\beta}$ 表示 $(\alpha,\beta)$ 块或子矩阵. 利用此记号, 块 $\boldsymbol{A}_{\alpha\beta}$ 是 $m_\alpha\times n_\beta$ 矩阵, 我们称 $\boldsymbol{A}=(\boldsymbol{A}_{\alpha\beta})$ 是 $q\times r$ 分块矩阵.

### 1.3.2 分块矩阵乘法

只要某些维数条件满足的话, 分块矩阵的组成就和具有标量元素的普通矩阵完全一样. 例如, 设

$$\boldsymbol{B}=\begin{array}{c}\left[\begin{array}{ccc}\boldsymbol{B}_{11} & \cdots & \boldsymbol{B}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{B}_{q1} & \cdots & \boldsymbol{B}_{qr}\end{array}\right]\begin{array}{c}m_1\\ \\ m_q\end{array}\\ \begin{array}{ccc}n_1 & \quad & n_r\end{array}\end{array},$$

我们就说 $\boldsymbol{B}$ 的分块与上面的 $\boldsymbol{A}$ 是"匹配的". 矩阵 $\boldsymbol{C}=\boldsymbol{A}+\boldsymbol{B}$ 也可看成是 $q\times r$ 分块矩阵:

$$\boldsymbol{C}=\left[\begin{array}{ccc}\boldsymbol{C}_{11} & \cdots & \boldsymbol{C}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{C}_{q1} & \cdots & \boldsymbol{C}_{qr}\end{array}\right]=\left[\begin{array}{ccc}\boldsymbol{A}_{11}+\boldsymbol{B}_{11} & \cdots & \boldsymbol{A}_{1r}+\boldsymbol{B}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{A}_{q1}+\boldsymbol{B}_{q1} & \cdots & \boldsymbol{A}_{qr}+\boldsymbol{B}_{qr}\end{array}\right].$$

分块矩阵的乘法稍微复杂. 我们先给出两个引理.

**引理 1.3.1** 设 $\boldsymbol{A}\in\mathbb{R}^{m\times p}$, $\boldsymbol{B}\in\mathbb{R}^{p\times n}$,

$$A = \begin{bmatrix} A_1 \\ \vdots \\ A_q \end{bmatrix} \begin{matrix} m_1 \\ \\ m_q \end{matrix} \qquad B = \underset{\qquad n_1 \qquad\quad n_r}{[B_1, \cdots, B_r]},$$

则

$$AB = C = \underset{\quad n_1 \qquad\qquad n_r}{\begin{bmatrix} C_{11} & \cdots & C_{1r} \\ \vdots & & \vdots \\ C_{q1} & \cdots & C_{qr} \end{bmatrix}} \begin{matrix} m_1 \\ \\ m_q \end{matrix}$$

其中$C_{\alpha\beta} = A_\alpha B_\beta (\alpha = 1 : q, \beta = 1 : r)$.

**证明** 首先我们比较 $C_{\alpha\beta}$ 中的元素与 $C$ 中元素的关系. 对 $1 \leqslant \alpha \leqslant q$, $1 \leqslant \beta \leqslant r, 1 \leqslant i \leqslant m_\alpha$ 和 $1 \leqslant j \leqslant n_\beta$, 我们有

$$[C_{\alpha\beta}]_{ij} = c_{\lambda+i,\mu+j},$$

其中

$$\lambda = m_1 + \cdots + m_{\alpha-1}, \quad \mu = n_1 + \cdots + n_{\beta-1}.$$

但是

$$c_{\lambda+i,\mu+j} = \sum_{k=1}^{p} a_{\lambda+i,k} b_{k,\mu+j} = \sum_{k=1}^{p} [A_\alpha]_{ik} [B_\beta]_{kj} = [A_\alpha B_\beta]_{ij}.$$

故知, $C_{\alpha\beta} = A_\alpha B_\beta$. □

**引理 1.3.2** 设 $A \in \mathbb{R}^{m\times p}$, $B \in \mathbb{R}^{p\times n}$,

$$A = \underset{\quad p_1 \qquad\quad p_s}{[A_1, \cdots, A_s]}, \qquad B = \begin{bmatrix} B_1 \\ \vdots \\ B_s \end{bmatrix} \begin{matrix} p_1 \\ \\ p_s \end{matrix},$$

则

$$AB = C = \sum_{\gamma=1}^{s} A_\gamma B_\gamma.$$

**证明** 我们令 $s = 2$ 并且把一般的 $s$ 留给读者去证明 (见习题 1.3.6). 对 $1 \leqslant i \leqslant m$ 和 $1 \leqslant j \leqslant n$, 我们有

$$\begin{aligned} c_{ij} &= \sum_{k=1}^{p} a_{ik} b_{kj} = \sum_{k=1}^{p_1} a_{ik} b_{kj} + \sum_{k=p_1+1}^{p_1+p_2} a_{ik} b_{kj} \\ &= [A_1 B_1]_{ij} + [A_2 B_2]_{ij} = [A_1 B_1 + A_2 B_2]_{ij}. \end{aligned}$$

故知, $C = A_1 B_1 + A_2 B_2$. □

对于一般的分块矩阵的相乘, 我们有如下结果.

**定理 1.3.3** 如果

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_{11} & \cdots & \boldsymbol{A}_{1s}\\ \vdots & & \vdots\\ \boldsymbol{A}_{q1} & \cdots & \boldsymbol{A}_{qs}\end{bmatrix}\begin{matrix}m_1\\ \\ m_q\end{matrix},\qquad \boldsymbol{B}=\begin{bmatrix}\boldsymbol{B}_{11} & \cdots & \boldsymbol{B}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{B}_{s1} & \cdots & \boldsymbol{B}_{sr}\end{bmatrix}\begin{matrix}p_1\\ \\ p_s\end{matrix},$$
$$\quad p_1 \qquad\quad p_s \qquad\qquad\qquad n_1 \qquad\quad n_r$$

而且我们把 $\boldsymbol{C}=\boldsymbol{AB}$ 按如下方式分块:

$$\boldsymbol{C}=\begin{bmatrix}\boldsymbol{C}_{11} & \cdots & \boldsymbol{C}_{1r}\\ \vdots & & \vdots\\ \boldsymbol{C}_{q1} & \cdots & \boldsymbol{C}_{qr}\end{bmatrix}\begin{matrix}m_1\\ \\ m_q\end{matrix},$$
$$n_1 \qquad\quad n_r$$

则

$$\boldsymbol{C}_{\alpha\beta}=\sum_{\gamma=1}^{s}A_{\alpha\gamma}B_{\gamma\beta},\quad \alpha=1:q,\beta=1:r.$$

**证明** 见习题 1.3.7. □

如果令 $q=2$, $s=2$, $r=1$ 和 $n_1=1$, 则得到一个很重要的特殊情形:

$$\begin{bmatrix}\boldsymbol{A}_{11} & \boldsymbol{A}_{12}\\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix}=\begin{bmatrix}\boldsymbol{A}_{11}\boldsymbol{x}_1+\boldsymbol{A}_{12}\boldsymbol{x}_2\\ \boldsymbol{A}_{21}\boldsymbol{x}_1+\boldsymbol{A}_{22}\boldsymbol{x}_2\end{bmatrix}.$$

这个分块的矩阵乘向量将在以后的章节中反复用到.

### 1.3.3 子矩阵记号

如同“普通”矩阵乘法一样, 分块矩阵乘法能用不同方式来组织. 我们需要引进一些记号来准确地刻画这些计算.

设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 且 $\boldsymbol{i}=(i_1,\cdots,i_r)$ 和 $\boldsymbol{j}=(j_1,\cdots,j_c)$ 是整数向量, 满足

$$i_1,\cdots,i_r\in\{1,2,\cdots,m\},\quad j_1,\cdots,j_c\in\{1,2,\cdots,n\}.$$

我们用 $A(i,j)$ 表示 $r\times c$ 子矩阵

$$A(i,j)=\begin{bmatrix}A(i_1,j_1) & \cdots & A(i_1,j_c)\\ \vdots & & \vdots\\ A(i_r,j_1) & \cdots & A(i_r,j_c)\end{bmatrix}.$$

如果下标向量 $i$ 和 $j$ 中的元素是连续的, 则可用 $\boldsymbol{A}$ 中的标量元素以及“冒号”记号来定义 $A(i,j)$. 特别地, 设 $1\leqslant i_1\leqslant i_2\leqslant m$ 和 $1\leqslant j_1\leqslant j_2\leqslant n$, 则 $A(i_1:i_2,j_1:j_2)$ 是把第 $i_1$ 行到第 $i_2$ 行和第 $j_1$ 列到第 $j_2$ 列取出来组成的矩阵, 例如

$$A(3:5,1:2)=\begin{bmatrix}a_{31} & a_{32}\\ a_{41} & a_{42}\\ a_{51} & a_{52}\end{bmatrix}.$$

关于子矩阵, 从 1.1.8 节知, 当 $i$ 和 $j$ 是标量时, $A(i,:)$ 表示 $\boldsymbol{A}$ 的第 $i$ 行, $A(:,j)$ 表示 $\boldsymbol{A}$ 的第 $j$ 列.

### 1.3.4 分块矩阵乘向量

定理 1.3.3 包括了一个重要的情形, 即分块矩阵乘以向量. 我们考虑 gaxpy 运算 $\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{y}$ 的详细过程, 其中 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, $\boldsymbol{x}\in\mathbb{R}^{n}$, $\boldsymbol{y}\in\mathbb{R}^{m}$ 且

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1\\ \vdots\\ \boldsymbol{A}_q\end{bmatrix}\begin{matrix}m_1\\ \\ m_q\end{matrix}\qquad \boldsymbol{y}=\begin{bmatrix}\boldsymbol{y}_1\\ \vdots\\ \boldsymbol{y}_q\end{bmatrix}\begin{matrix}m_1\\ \\ m_q\end{matrix}.$$

我们称 $\boldsymbol{A}_i$ 为第 $i$ 个块行. 如果 m.vec=$(m_1,\cdots,m_q)$ 是块行之“高度”的向量, 则从

$$\begin{bmatrix}\boldsymbol{y}_1\\ \vdots\\ \boldsymbol{y}_q\end{bmatrix}=\begin{bmatrix}\boldsymbol{A}_1\\ \vdots\\ \boldsymbol{A}_q\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}\boldsymbol{y}_1\\ \vdots\\ \boldsymbol{y}_q\end{bmatrix},$$

我们得到

```
last=0
for i = 1 : q
    first = last+1
    last = first+m.vec(i) − 1                                  (1.3.1)
    y(first:last) = A(first : last, :)x + y(first : last)
end
```

每次循环中算法执行的是一个“普通”的 gaxpy, 可用算法 1.1.3 或者算法 1.1.4 计算.

另一种将 gaxpy 计算分块的方式是将 $\boldsymbol{A}$ 和 $\boldsymbol{x}$ 以如下方式划分:

$$\boldsymbol{A}=\begin{matrix}[\boldsymbol{A}_1 & \cdots & \boldsymbol{A}_r]\\ n_1 & & n_r\end{matrix},\quad \boldsymbol{x}=\begin{bmatrix}\boldsymbol{x}_1\\ \vdots\\ \boldsymbol{x}_r\end{bmatrix}\begin{matrix}n_1\\ \\ n_r.\end{matrix}$$

这时, 我们称 $\boldsymbol{A}_i$ 是 $\boldsymbol{A}$ 的第 $j$ 个块列. 设 n.vec $=(n_1,\cdots,n_r)$ 是块列之宽度的向量, 则从

$$\boldsymbol{y}=[\boldsymbol{A}_1,\cdots,\boldsymbol{A}_r]\begin{bmatrix}\boldsymbol{x}_1\\ \vdots\\ \boldsymbol{x}_r\end{bmatrix}+\boldsymbol{y}=\sum_{j=1}^{r}\boldsymbol{A}_j\boldsymbol{x}_j+\boldsymbol{y}$$

得到

```
last=0
```

$$
\begin{aligned}
&\textbf{for } j = 1:r \\
&\quad \text{first=last+1} \\
&\quad \text{last=first+n.vec}(j) - 1 \\
&\quad y = A(:, \text{first}:\text{last})x(\text{first}:\text{last}) + y \\
&\textbf{end}
\end{aligned}
\tag{1.3.2}
$$

同样地, 循环中的每次 gaxpy 计算可利用算法 1.1.3 或者算法 1.1.4.

### 1.3.5 分块矩阵乘法

和普通的以标量为元素的矩阵乘法一样, 分块矩阵乘法也可写成不同形式. $\boldsymbol{A}$, $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 的不同分块方式可为导出 1.1 节中的点积、saxpy 以及外积算法之分块形式提供条件. 为了用尽可能少的下标来说明这一点, 我们假定这 3 个矩阵都是 $n \times n$ 矩阵且 $n = Nl$, $N$ 和 $l$ 都是正整数.

设 $\boldsymbol{A} = (\boldsymbol{A}_{\alpha\beta}), \boldsymbol{B} = (\boldsymbol{B}_{\alpha\beta})$ 和 $\boldsymbol{C} = (\boldsymbol{C}_{\alpha\beta})$ 都是 $l \times l$ 块的 $N \times N$ 分块矩阵, 从定理 1.3.3 可知

$$
\boldsymbol{C}_{\alpha\beta} = \sum_{\gamma=1}^{N} \boldsymbol{A}_{\alpha\gamma}\boldsymbol{B}_{\gamma\beta} + \boldsymbol{C}_{\alpha\beta}, \quad \alpha = 1:N, \quad \beta = 1:N.
$$

如果我们按这一求和公式组织矩阵乘法, 则得到算法 1.1.5 的分块形式

$$
\begin{aligned}
&\textbf{for } \alpha = 1:N \\
&\quad i = (\alpha - 1)l + 1 : \alpha l \\
&\quad \textbf{for } \beta = 1:N \\
&\quad\quad j = (\beta - 1)l + 1 : \beta l \\
&\quad\quad \textbf{for } \gamma = 1:N \\
&\quad\quad\quad k = (\gamma - 1)l + 1 : \gamma l \\
&\quad\quad\quad C(i,j) = A(i,k)B(k,j) + C(i,j) \\
&\quad\quad \textbf{end} \\
&\quad \textbf{end} \\
&\textbf{end}
\end{aligned}
\tag{1.3.3}
$$

注意到在 $l = 1$ 时有 $\alpha \equiv i$, $\beta \equiv j$ 和 $r \equiv k$, 就回到算法 1.1.5.

为了得到一个分块的 saxpy 矩阵乘法, 我们把 $\boldsymbol{C} = \boldsymbol{AB} + \boldsymbol{C}$ 写成

$$
[\boldsymbol{C}_1, \cdots, \boldsymbol{C}_N] = [\boldsymbol{A}_1, \cdots, \boldsymbol{A}_N] \begin{bmatrix} \boldsymbol{B}_{11} & \cdots & \boldsymbol{B}_{1N} \\ \vdots & \ddots & \vdots \\ \boldsymbol{B}_{N1} & \cdots & \boldsymbol{B}_{NN} \end{bmatrix} + [\boldsymbol{C}_1, \cdots, \boldsymbol{C}_N],
$$

其中 $\boldsymbol{A}_\alpha, \boldsymbol{C}_\alpha \in \mathbb{R}^{n \times l}, \boldsymbol{B}_{\alpha\beta} \in \mathbb{R}^{l \times l}$. 从而可得到

$$
\begin{aligned}
&\textbf{for } \beta = 1:N \\
&\quad j = (\beta - 1)l + a : \beta l
\end{aligned}
$$

$$
\begin{aligned}
&\quad \textbf{for } \alpha = 1:N \\
&\qquad i = (\alpha-1)l+1:\alpha l \\
&\qquad C(:,j) = A(:,i)B(i,j) + C(:,j) \\
&\quad \textbf{end} \\
&\textbf{end}
\end{aligned}
\tag{1.3.4}
$$

此算法是算法 1.1.7 的分块形式.

分块的外积算法可由下面划分来导出:

$$
\boldsymbol{A} = [\boldsymbol{A}_1, \cdots, \boldsymbol{A}_N], \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{B}_N^{\mathrm{T}} \end{bmatrix},
$$

其中 $\boldsymbol{A}_\gamma, \boldsymbol{B}_\gamma \in \mathbb{R}^{n\times l}$. 从引理 1.3.2 可知

$$
\boldsymbol{C} = \sum_{\gamma=1}^{N} \boldsymbol{A}_\gamma \boldsymbol{B}_\gamma^{\mathrm{T}} + \boldsymbol{C},
$$

故得到

$$
\begin{aligned}
&\textbf{for } \gamma = 1:N \\
&\qquad k = (\gamma-1)l+1:\gamma l \\
&\qquad C = A(:,k)B(k,:) + C \\
&\textbf{end}
\end{aligned}
\tag{1.3.5}
$$

这就是算法 1.1.8 的分块形式.

### 1.3.6 复矩阵乘法

考虑复矩阵乘法:

$$
\boldsymbol{C}_1 + \mathrm{i}\boldsymbol{C}_2 = (\boldsymbol{A}_1 + \mathrm{i}\boldsymbol{A}_2)(\boldsymbol{B}_1 + \mathrm{i}\boldsymbol{B}_2) + (\boldsymbol{C}_1 + \mathrm{i}\boldsymbol{C}_2),
$$

其中所有的矩阵都是实的, $\mathrm{i}^2 = -1$. 比较实部和虚部就得到

$$
\boldsymbol{C}_1 = \boldsymbol{A}_1\boldsymbol{B}_1 - \boldsymbol{A}_2\boldsymbol{B}_2 + \boldsymbol{C}_1, \quad \boldsymbol{C}_2 = \boldsymbol{A}_1\boldsymbol{B}_2 + \boldsymbol{A}_2\boldsymbol{B}_1 + \boldsymbol{C}_2,
$$

也可表示成矩阵形式

$$
\begin{bmatrix} \boldsymbol{C}_1 \\ \boldsymbol{C}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_1 & -\boldsymbol{A}_2 \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 \end{bmatrix} \begin{bmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}_1 \\ \boldsymbol{C}_2 \end{bmatrix}.
$$

这表明可以用实矩阵软件来求解复矩阵问题. 唯一不足的是在矩阵

$$
\widetilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A}_1 & -\boldsymbol{A}_2 \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 \end{bmatrix}
$$

中需要将矩阵 $\boldsymbol{A}_1$ 和 $\boldsymbol{A}_2$ 储存两次.

### 1.3.7 “分而治之”矩阵乘法

我们以讨论矩阵乘法完全不同的方式来结束本节. 首先讨论 $2\times 2$ 块矩阵的乘法:

$$\begin{bmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{12} \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} \end{bmatrix},$$

其中每块都是方阵. 在普通算法中, $\boldsymbol{C}_{ij} = \boldsymbol{A}_{i1}\boldsymbol{B}_{1j} + \boldsymbol{A}_{i2}\boldsymbol{B}_{2j}$. 一共有 8 次乘法和 4 次加法. Strassen(1969) 证明仅用 7 次乘法和 18 次加减法可计算 $\boldsymbol{C}$:

$$\begin{aligned}
\boldsymbol{P}_1 &= (\boldsymbol{A}_{11} + \boldsymbol{A}_{22})(\boldsymbol{B}_{11} + \boldsymbol{B}_{22}),\\
\boldsymbol{P}_2 &= (\boldsymbol{A}_{21} + \boldsymbol{A}_{22})\boldsymbol{B}_{11},\\
\boldsymbol{P}_3 &= \boldsymbol{A}_{11}(\boldsymbol{B}_{12} - \boldsymbol{B}_{22}),\\
\boldsymbol{P}_4 &= \boldsymbol{A}_{22}(\boldsymbol{B}_{21} - \boldsymbol{B}_{11}),\\
\boldsymbol{P}_5 &= (\boldsymbol{A}_{11} + \boldsymbol{A}_{12})\boldsymbol{B}_{22},\\
\boldsymbol{P}_6 &= (\boldsymbol{A}_{21} - \boldsymbol{A}_{11})(\boldsymbol{B}_{11} + \boldsymbol{B}_{12}),\\
\boldsymbol{P}_7 &= (\boldsymbol{A}_{12} - \boldsymbol{A}_{22})(\boldsymbol{B}_{21} + \boldsymbol{B}_{22}),\\
\boldsymbol{C}_{11} &= \boldsymbol{P}_1 + \boldsymbol{P}_4 - \boldsymbol{P}_5 + \boldsymbol{P}_7,\\
\boldsymbol{C}_{12} &= \boldsymbol{P}_3 + \boldsymbol{P}_5,\\
\boldsymbol{C}_{21} &= \boldsymbol{P}_2 + \boldsymbol{P}_4,\\
\boldsymbol{C}_{22} &= \boldsymbol{P}_1 + \boldsymbol{P}_3 - \boldsymbol{P}_2 + \boldsymbol{P}_6,
\end{aligned}$$

直接替换即可证明这些公式. 假定 $n = 2m$, 则每块都是 $m\times m$. 计算 $\boldsymbol{C} = \boldsymbol{AB}$ 中加法和乘法次数, 可知常规矩阵乘法需要 $(2m)^3$ 次乘法和 $(2m)^3-(2m)^2$ 次加减法. 而如果在分块矩阵的常规乘法中用 Strassen 算法, 则需要 $7m^3$ 次乘法和 $7m^3+11m^2$ 次加减法. 如果 $m \gg 1$, 则 Strassen 算法所需运算量是常规算法的 7/8.

可以看出, Strassen 思想能反复应用. 特别地, 我们可在求每个 $\boldsymbol{P}_i$ 的一半维数的块矩阵乘法中用 Strassen 算法. 所以, 当 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是 $n\times n$ 矩阵且 $n = 2^q$ 时, 我们可重复应用 Strassen 算法. 在最底层, 块的大小是 $1\times 1$. 当然, 也没必要一直递推到 $n = 1$. 当块的维数足够小 $(n \leqslant n_{\min})$ 时, 计算 $P_i$ 就可用常规的矩阵乘法. 下面是一个完整的算法.

**算法 1.3.1 (Strassen 乘法)** 设 $n = 2^q$, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 $\boldsymbol{B} \in \mathbb{R}^{n\times n}$. 如果 $n_{\min} = 2^d (d \leqslant q)$, 则本算法递归应用 Strassen 技巧 $q-d$ 次计算 $\boldsymbol{C} = \boldsymbol{AB}$.

**function**: $C = \textbf{strass}(A, B, n, n_{\min})$

  **if** $n \leqslant n_{\min}$

    $C = AB$

  **else**

    $m = n/2; u = 1:m; v = m+1:n;$

    $P_1 = \textbf{strass}(A(u,u) + A(v,v), B(u,u) + B(v,v), m, n_{\min})$

$P_2 = \textbf{strass}(A(v,u)+A(v,v), B(u,u), m, n_{\min})$
$P_3 = \textbf{strass}(A(u,u), B(u,v)-B(v,v), m, n_{\min})$
$P_4 = \textbf{strass}(A(v,v), B(v,u)-B(u,u), m, n_{\min})$
$P_5 = \textbf{strass}(A(u,u)+A(u,v), B(v,v), m, n_{\min})$
$P_6 = \textbf{strass}(A(v,u)-A(u,u), B(u,u)+B(u,v), m, n_{\min})$
$P_7 = \textbf{strass}(A(u,v)-A(v,v), B(v,u)+B(v,v), m, n_{\min})$
$C(u,u) = P_1 + P_4 - P_5 + P_7$
$C(u,v) = P_3 + P_5$
$C(v,u) = P_2 + P_4$
$C(v,v) = P_1 + P_3 - P_2 + P_6$

**end**

与我们以前所介绍的算法不同,"strass"是递归的,它在程序中调用自身."分而治之"类型的算法常常用这种方式叙述,我们把这个算法写成 MATLAB 函数,从而递归调用可精确地写出.

**strass** 算法的运算量是 $n$ 和 $n_{\min}$ 的复杂函数. 当 $n_{\min} \gg 1$ 时,由于加法的次数与乘法的次数大体相同,所以只需计算乘法次数. 而统计乘法运算次数只需分析递归的最深层,因为只有在此层才有乘法运算. **strass** 要重复调用 $q-d$ 次,故需要 $7^{q-d}$ 个常规的矩阵乘法. 这些乘法中矩阵的维数是 $n_{\min}$,故 **strass** 算法所需的乘法次数为 $s=(2^d)^3 \times 7^{q-d}$. 常规的矩阵乘法中乘法运算次数为 $c=(2^q)^3$. 注意到

$$\frac{s}{c} = \left(\frac{2^d}{2^q}\right)^3 7^{q-d} = \left(\frac{7}{8}\right)^{q-d}.$$

如果 $d=0$,即我们递归到 $1\times 1$ 块,则

$$s = \left(\frac{7}{8}\right)^q c = 7^q = n^{\log_2 7} \approx n^{2.807}.$$

所以,在渐近意义上,strass 方法的乘法次数是 $O(n^{2.807})$. 但是,当 $n_{\min}$ 很小时,加法的次数 (与乘法次数相比) 将会是很大.

**例 1.3.1** *如果 $n=1024$ 和 $n_{\min}=64$,则 strass 算法需要的运算量仅是常规方法的 $(7/8)^{10-6} \approx 0.6$.*

## 习 题

**1.3.1** 推广 (1.3.3) 使其能处理划分是定理 1.3.3 所给出的情形.

**1.3.2** 推广 (1.3.4) 和 (1.3.5) 使之能处理不同的维数的分块.

**1.3.3** 修改 strass 使之可应用于任何维数的方阵乘法. 提示: 如果"当前"的 $\boldsymbol{A}$ 是奇数维,则加一个零行和零列.

**1.3.4** 如果

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1r} \\ \vdots & \ddots & \vdots \\ A_{q1} & \cdots & A_{qr} \end{bmatrix}$$

是 $A$ 的分块, 证明

$$A^{\mathrm{T}} = \begin{bmatrix} A_{11}^{\mathrm{T}} & \cdots & A_{q1}^{\mathrm{T}} \\ \vdots & \ddots & \vdots \\ A_{1r}^{\mathrm{T}} & \cdots & A_{qr}^{\mathrm{T}} \end{bmatrix}.$$

**1.3.5** 设 $n$ 是偶数, 定义从 $\mathbb{R}^n$ 到 $\mathbb{R}$ 的函数

$$f(x) = x(1:2:n)^{\mathrm{T}} x(2:n) = \sum_{i=1}^{n/2} x_{2i-1} x_{2i}.$$

(a) 证明对任何 $x, y \in \mathbb{R}^n$ 有

$$x^{\mathrm{T}} y = \sum_{i=1}^{n/2} (x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1}) - f(x) - f(y).$$

(b) 考虑 $n \times n$ 矩阵相乘 $C = AB$, 给出一个通过将 $f$ 应用到 $A$ 的行以及 $B$ 的列的计算乘积 $C = AB$ 的需要 $n^3/2$ 次乘法的算法. 详细讨论可见 Winograd(1968).

**1.3.6** 对一般的 $s$ 证明引理 1.3.2. 提示: 令 $\rho_1 = 0$,

$$\rho_\gamma = p_1 + \cdots + p_{\gamma-1}, \quad \gamma = 1 : s+1,$$

证明

$$c_{ij} = \sum_{\gamma=1}^{s} \sum_{k=\rho_\gamma+1}^{\rho_\mu+1} a_{ik} b_{kj}.$$

**1.3.7** 利用引理 1.3.1 和引理 1.3.2 证明定理 1.3.3. 特别地, 取

$$A_\gamma = \begin{bmatrix} A_{1\gamma} \\ \vdots \\ A_{q\gamma} \end{bmatrix}, \quad B_\gamma = [B_{\gamma 1}, \cdots, B_{\gamma r}],$$

从引理 1.3.2 知

$$C = \sum_{\gamma=1}^{s} A_\gamma B_\gamma.$$

借助于引理 1.3.1 分析每一个 $A_\gamma B_\gamma$.

## 本节注释与参考文献

很长时间以来, 矩阵乘法的快速算法在计算机科学中受到高度重视, 请参阅:

S.Winograd(1968). "A New Algorithm for Inner Product," *IEEE Trans. Comp.* C-17, 693–694.

V. Strassen(1969). "Gaussian Elimination is Not Optimal," *Numer. Math.* 13, 354–356.

V. Pan(1984). "How Can We Speed Up Matrix Multiplication?," *SIAM Review* 26, 393–416.

许多此类方法的实际价值并不清楚. 但是在

D. Bailey(1988). "Extra High Speed Matrix Multiplication on the Cray-2," *SIAM J. Sci. and Stat. Comp.* 9, 603–607.

一文发表之后, 很明显地知道全盘否定这类快速方法是不明智的. Strass 算法的"稳定性"将在 2.4.10 节中讨论. 也可参阅:

N. J. Higham(1990). "Exploiting Fast Matrix Multiplication within the Level 3 BLAS," *ACM Trans. Math. Soft.* 16, 352–368.

C. C. Douglas, M. Heroux, G. Slishman, and R. M. Smith(1994). "GEMMW: A Portable Level 3 BLAS Winograd Variant of Strassen's Matrix-Matrix Multiply Algorithm," *J. Comput. Phys.* 110, 1–10.

## 1.4 向量化与数据重复使用

本书中讨论的矩阵乘法大多是基于点积和 saxpy 运算. 向量流水线计算机能很快执行这类向量运算, 因为它的硬件充分利用向量运算是一连串的标量运算这一特点. 这样的计算机效率是否高取决于向量的长度以及其他一些关于数据移动的因素, 如向量间、向量存储的次数以及数据重新利用的程度等. 我们的目的是熟悉这些因素. 我们并不试图设计可用来预估表现的完整的向量计算模型. 我们只想指出那些在设计有效的向量计算程序中的思路. 我们不考虑任何特定的计算机. 关于具体机型的讨论有大量的参考文献说明.

### 1.4.1 流水线运算

向量计算机快的主要原因是与"流水线"有关. 流水线的概念最好是用生产装配线作为例子来说明. 假定组装每辆汽车需要在装配线上的 60 个工作台每一台花 1 分钟. 如果组装线上人员充足, 每分钟都能启动一次新的组装, 那么组装 1000 辆汽车大约需要 1000+60=1060 分钟. 对于这种生产量来说, 该条生产线的有效"向量速度"是每分钟 1000/1060 辆. 在另一方面, 如果组装线上人员不足, 则要 1 小时才能启动一次新的组装, 所以制造 1000 辆汽车需要 1000 小时. 在这种情况下, 这条生产线的有效"标量速度"是每分钟 1/60 辆.

同样, 对向量相加 $z = x + y$ 这样的流水线向量运算也是如此. 每个标量运算 $z_i = x_i + y_i$ 像是一辆车, 向量的元素个数就如生产量. 如果每个 $z_i$ 从开始到结束的时间是 $\tau$, 则利用流水线的 $n$ 维向量相加可以在远小于 $n\tau$ 的时间完成, 这就产生了向量速度. 没有流水线, 向量计算以标量速度进行, 从而将需要大约 $n\tau$ 时间完成.

我们看一组浮点运算怎样才可以流水线化. 浮点运算通常需要几步完成. 举例来说, 两个标量 $x$ 和 $y$ 相加的 3 步骤可按图 1.4.1 进行. 为说明这一运算, 我们还用上面的比喻, 把加法器看成一条有三个"工作台"的装配线. 输入的标量 $x$ 和 $y$ 要在三个工作台上都花上一步, 三步之后和 $z$ 就出来了. 注意到, "无等待"的单个

加法需要计算时, 在计算过程中三个工作台仅有当中之一个是忙的.

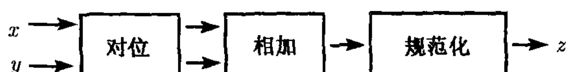



图 1.4.1 3 个步骤的加法器

现在, 我们考虑向量相加 $z=x+y$. $x$ 和 $y$ 就像在流水线上似的通过加法器. 一旦流水线满载并达到稳定状态后, 每一个循环就可得到一个 $z_i$. 图 1.4.2 刻画了流水线满载后的情形. 在这种情形. 向量速度就大约是标量速度的三倍, 这是由于每一个单独的加法需要 3 步.

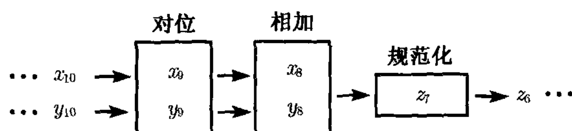



图 1.4.2 流水线化相加

### 1.4.2 向量运算

向量计算机具有"向量指令"的功能, 如向量相加、向量乘法、向量加数、点积和 saxpy. 为明了起见, 我们假定是在"向量寄存"下考虑这些运算. 向量在寄存器和存储器之间可通过"vector load"和"vector store"指令来交换.

向量处理器中的一个重要的因素是向量寄存器的长度, 我们记之为 $v_L$. 一个 $n$ 维向量运算必须分为若干个长度不超过 $v_L$ 的子向量的运算. 以下就 $n$ 维向量相加 $z=x+y$ 来说明如何进行划分.

```
first=1
while first≤ n
        last=min{n, first + v_L − 1}
        vector load x(first:last).
        vector load y(first:last).
        vector add: z(first:last)=x(first:last)+y(first:last).
        vector store z(first:last).
        first=last+1
end
```

向量计算机的合理的编译器应该可以从程序中的 $z=x+y$ 指令自动产生上述向量指令.

### 1.4.3 向量长度

假定向量运算 $op$ 的流水线需要 $\tau_{op}$ 步"启动". 还假定当流水线满载后每步

可得到结果的一个分量. 完成一个 $n$ 维的 $op$ 所需时间为

$$T_{op}(n) = (\tau_{op} + n)\mu, \quad n \leqslant v_L,$$

其中 $\mu$ 是每步所需时间, $v_L$ 是向量硬件的长度.

如果要考虑的向量长于向量硬件的长度, 我们已看到必须将整体的向量运算截成硬件能处理的分段运算. 所以, 当

$$n = n_1 v_L + n_0, \quad 0 \leqslant n_0 < v_L$$

时, 我们可假定

$$T_{op}(n) = \begin{cases} n_1(\tau_{op} + v_L)\mu, & n_0 = 0 \\ (n_1(\tau_{op} + v_L) + \tau_{op} + n_0)\mu, & n_0 \neq 0 \end{cases}$$

是 $n$ 维运算 $op$ 所需的全部时间, 上式可简化成

$$T_{op}(n) = (n + \tau_{op}\text{ceil}(n/v_L))\mu,$$

其中 $\text{ceil}(\alpha)$ 是满足于 $\alpha \leqslant \text{ceil}(\alpha)$ 的最小整数. 假定计算每个分量需要 $\rho$ 个 flop, 则对一般的 $n$, 计算的效率是

$$R_{op}(n) = \frac{\rho n}{T_{op}(n)} = \frac{\rho}{\mu} \frac{1}{1 + \frac{\tau_{op}}{n}\text{ceil}\left(\frac{n}{v_L}\right)}.$$

(如果 $\mu$ 的单位是秒, 则 $R_{op}$ 是每秒的 flop 数.) 渐近效率是

$$\lim_{n\to\infty} R_{op}(n) = \frac{1}{1 + \frac{\tau_{op}}{v_L}} \frac{\rho}{\mu}.$$

作为评价向量运算中启动开销之影响的程度的一种方式, Hockney and Jesshope (1988) 定义 $n_{1/2}$ 为达到一半峰值的最小的 $n$, 即

$$\frac{\rho n_{1/2}}{T_{op}(n_{1/2})} = \frac{1}{2}\frac{\rho}{\mu}.$$

具有大因子 $n_{1/2}$ 的计算机执行短向量运算时效果不好.

我们现在看以上计算模型如何运用到矩阵乘法修正 $\boldsymbol{C} = \boldsymbol{AB} + \boldsymbol{C}$, 其中 $\boldsymbol{A} \in \mathbb{R}^{m\times p}$, $\boldsymbol{B} \in \mathbb{R}^{p\times n}$ 和 $\boldsymbol{C} \in \mathbb{R}^{m\times n}$. 从 1.1.11 节可知有六种可能的常规算法, 它们对应于下面循环的六种可能排序:

```
for i = 1 : m
    for j = 1 : n
        for  k = 1 : p
            C(i, j) = A(i, k)B(k, j) + C(i, j)
        end
```

```
    end
  end
```

这是 $ijk$ 形式, 它的最内层循环是 $p$ 维向量的点积. 因而, 从我们的效率模型可知需要

$$T_{ijk} = mnp + mn \cdot \text{ceil}(p/v_L)\tau_{dot}$$

步. 对其他几种形式进行同样分析可得下表.

| 形 式 | 步 |
|---|---|
| $ijk$ | $mnp + mn \cdot \text{ceil}(p/v_L) \cdot \tau_{dot}$ |
| $jik$ | $mnp + mn \cdot \text{ceil}(p/v_L) \cdot \tau_{dot}$ |
| $ikj$ | $mnp + mp \cdot \text{ceil}(n/v_L) \cdot \tau_{sax}$ |
| $jki$ | $mnp + np \cdot \text{ceil}(m/v_L) \cdot \tau_{sax}$ |
| $kij$ | $mnp + mp \cdot \text{ceil}(n/v_L) \cdot \tau_{sax}$ |
| $kji$ | $mnp + np \cdot \text{ceil}(m/v_L) \cdot \tau_{sax}$ |

我们基于简单的整数计数进行一些观察. 假定 $\tau_{sax}$ 和 $\tau_{dot}$ 大致相等. 如果 $m, n$ 和 $p$ 都小于 $v_L$, 则最有效的形式应具有最长的内层循环. 如果 $m, n$ 和 $p$ 都远大于 $v_L$, 则六种形式的差别甚微.

### 1.4.4 "间"的概念

向量运算在存储上的花费对运行速度是有影响的. 主要的因素是"间". 储存浮点向量的"间"是向量元素 (在逻辑内存位置) 之间的距离. 读取两维 Fortran 数组的行不是整体间运算, 因为数组是按列储存的. 与此相反, 在 C 语言中, 矩阵是按行储存的. 非整体间运算影响计算机的流水线能力, 从而降低效率.

为说明"间"的作用我们考虑矩阵乘法的六种形式在最内层是如何从矩阵 $\boldsymbol{A}, \boldsymbol{B}$ 和 $\boldsymbol{C}$ 中"抽取"数据的. 这是向量计算 (点积或 saxpy) 所在, 一共有三种可能性:

$jki$ 或 $kji$ **for** $i = 1:m$

$$C(i,j) = C(i,j) + A(i,k)B(k,j)$$

**end**

$ikj$ 或 $kij$ **for** $j = 1:n$

$$C(i,j) = C(i,j) + A(i,k)B(k,j)$$

**end**

$ijk$ 或 $jik$ **for** $k = 1:p$

$$C(i,j) = C(i,j) + A(i,k)B(k,j)$$

**end**

下表是这三种情形所对应的 $\boldsymbol{A}, \boldsymbol{B}$ 以及 $\boldsymbol{C}$ 的"间":

| 形 式 | $A$ 的"间" | $B$ 的"间" | $C$ 的"间" |
|---|---|---|---|
| $jki$ 或 $kji$ | 整体 | 0 | 整体 |
| $ikj$ 或 $kij$ | 0 | 非整体 | 非整体 |
| $ijk$ 或 $jik$ | 非整体 | 整体 | 0 |

假定储存是按列为顺序的. 0"间"代表在内层循环中仅用到数据的一个元素. 从"间"的角度来看, 我们应该偏爱 $jki$ 和 $kji$ 形式. 这可能与基于向量长度所考虑的偏好不一致. 这种困境在高性能计算是有代表性的, 即一个目标 (极大化向量长度) 与另一个目标 (要求整体"间") 相矛盾.

有时, 整体间与向量长度的矛盾可通过适当地选取数据来解决. 考虑 gaxpy 运算 $y = Ax + y$, 其中 $A \in \mathbb{R}^{n\times n}$ 是对称矩阵. 为简单起见, 假定 $n \leqslant v_L$. 设 $A$ 是按常规储存且应用算法 1.1.4, 则核心的计算是 $n$ 个 $n$ 维向量的 saxpy:

```
for j = 1 : n
    y = A(:, j)x(j) + y
end
```

简单的运行模型表明需要

$$T_1 = n(\tau_{sax} + n)$$

步.

在 1.2.7 节我们介绍了对称矩阵的下三角储存方式且得到如下形式的 gaxpy:

```
for j = 1 : n
    for i = 1 : j - 1
        y(i) = A.vec((i - 1)n - i(i - 1)/2 + j)x(j) + y(i)
    end
    for i = j : n
        y(i) = A.vec((j - 1)n - j(j - 1)/2 + i)x(j) + y(i)
    end
end
```

注意到第一个 $i$ 循环并不是整体间的 saxpy. 如果我们假定一个 $n$ 维的非整体间 saxpy 等价于 $n$ 个整体间的 saxpy(一个最坏的情形). 则此算法需要

$$T_2 = n\left(\frac{n}{2}\tau_{sax} + n\right)$$

步.

在 1.2.8 节中我们给出了按对角储存的算法:

```
for i = 1 : n
    y(i) = A.diag(i)x(i) + y(i)
end
for k = 1 : n - 1
```

```
    t = nk - k(k-1)/2
    {y = D(A,k)x + y}
    for i = 1 : n-k
        y(i) = A.diag(i+t)x(i+k) + y(i)
    end
    {y = D(A,k)^T x + y}
    for i = 1 : n-k
        y(i+k) = A.diag(i+t)x(i) + y(i+k)
    end
end
```

在这种情形下, 每个内层循环都是整体间乘法 ($vm$), 根据我们的模型, 它需要

$$T_3 = n(2\tau_{vm} + n)$$

步.

此例说明了数据的结构可以影响一个算法之"间"的性质. 按对角储存看上去很有吸引力, 因为它把矩阵紧凑地表示且有整体间性质. 然而, 仔细地分析哪一种方式最好, 取决于 $\tau_{sax}$ 和 $\tau_{vm}$ 的值和非整体间计算以及过量储存所导致的精确影响. 这是一个复杂的问题, 它需要精心设计的标准检查程序.

### 1.4.5 考虑数据移动

矩阵算法中的另一个重要因素是关于在算法执行过程中需要移动的数据量. 矩阵储存于内存之中, 关于矩阵元素的计算在计算器中进行. 在许多计算机中内存通信的控制对性能是十分重要的. 继续用本节开始时所用的比喻:"我们能否足够快输送矩阵数据, 以保持超快的计算器是忙的? 我们能否把结果快速送回内存, 而不致于数据积压?" 图 1.4.3 给出一个典型的单处理器情形. 具体的设计在不同的机器中是不同的, 但有两条"公理"是满足的:

- 分级中的每级的储存有限且由于经济原因级越高储存就越小。
- 数据在两级储存之间的移动是需要费时的, 有时还是比较大的.

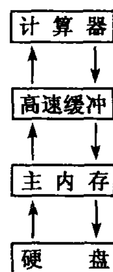



图 1.4.3 内存分级系统

设计一个有效的矩阵算法需要仔细考虑数据在不同级别存储的流动. 关于这方面, 向量"触"和数据再用是重要的问题.

### 1.4.6 向量触

在许多超级计算机中, 数据量分段 (例如, 向量) 是移动的. 从内存读写一组向量所需的时间大致相当于用这组向量进行点积式 saxpy 所需的时间, 因而, 矩阵程

序中向量触的数目是重要的统计数. 所谓"向量触", 指的是一次向量读取或向量存写.

我们数一下 $m \times n$ 外积的向量触数目. 假定 $m = m\ v_L$ 和 $n = n_1 v_L$, 其中 $v_L$ 是硬件的向量长度 (见 1.4.3 节). 在这种情况下, 外积修正 $\boldsymbol{A} = \boldsymbol{A} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}$ 可整理为

$$
\begin{aligned}
&\textbf{for } \alpha = 1 : m_1 \\
&\qquad i = (\alpha - 1)v_L + 1 : \alpha v_L \\
&\qquad \textbf{for } \beta = 1 : n_1 \\
&\qquad\qquad j = (\beta - 1)v_L + 1 : \beta v_L \\
&\qquad\qquad A(i,j) = A(i,j) + x(i)y(j)^{\mathrm{T}} \\
&\qquad \textbf{end} \\
&\textbf{end}
\end{aligned}
$$

子矩阵 $A(i,j)$ 的每一列都需要读取、修正然后存回去, 再加上 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 的向量触, 我们发现一共大约需要

$$
\sum_{\alpha=1}^{m} \left( 1 + \sum_{\beta=1}^{n_1} (1 + 2v_L) \right) \approx 2m_1 n
$$

向量触. (低阶项在分析中舍去了.)

现在考虑 gaxpy 修正 $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}$, 其中 $\boldsymbol{y} \in \mathbb{R}^m$, $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. 把计算分解为长度为 $v_L$ 的小段可得到

$$
\begin{aligned}
&\textbf{for } \alpha = 1 : m_1 \\
&\qquad i = (\alpha - 1)v_L + 1 : \alpha v_L \\
&\qquad \textbf{for } \beta = 1 : n_1 \\
&\qquad\qquad j = (\beta - 1)v_L + 1 : \beta v_L \\
&\qquad\qquad y(i) = y(i) + A(i,j)x(j) \\
&\qquad \textbf{end} \\
&\textbf{end}
\end{aligned}
$$

同样, 子矩阵 $A(i,j)$ 的每一列都需要读取, 但只需把与 $y$ 有关的子向量存回去. 所以 $m \times n$ gaxpy 的向量触数目为

$$
\sum_{\alpha=1}^{m_1} \left( 2 + \sum_{\beta=1}^{n_1} (1 + v_L) \right) \approx m_1 n.
$$

这是同样维数外积所需的向量触的一半. 所以, 当一个计算可写成外积或者是 gaxpy 时, 从向量触的角度来看最好是写成后者.

### 1.4.7 分块与再用

缓冲是介于运算器与主内存之间的一个小的高速内存, 见图 1.4.3. 缓冲的有效利用影响计算表现, 因为它直接关系到数据在运算器与下级内存的流动.

为说明这一点, 我们考虑矩阵乘法修正 $\boldsymbol{C}=\boldsymbol{AB}+\boldsymbol{C}$, 其中 $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}\in\mathbb{R}^{n\times n}$ 存于主内存[①]. 所有数据必须通过缓冲才能到达进行浮点运算的运算器. 如果缓冲小而 $n$ 较大, 则修正必须分成小的部分, 缓冲才可"体面地"让数据通过.

一种方式是将 $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 分块:

$$\boldsymbol{B}=[\underset{l}{\boldsymbol{B}_1},\cdots,\underset{l}{\boldsymbol{B}_N}],\qquad \boldsymbol{C}=[\underset{l}{\boldsymbol{C}_1},\cdots,\underset{l}{\boldsymbol{C}_N}],$$

其中 $n=lN$. 从展开式

$$\boldsymbol{C}_\alpha=\boldsymbol{AB}_\alpha+\boldsymbol{C}_\alpha=\sum_{k=1}^{n}A(:,k)B_\alpha(k,:)+\boldsymbol{C}_\alpha$$

我们得到如下计算框架

```
for α = 1 : N
        把 B_α 和 C_α 取到缓冲
        for k = 1 : n
                把 A(:,k) 取到缓冲并计算 C_α
                C_α = A(:,k)B_α(k,:) + C_α
        end
        把 C_α 存回主内存
end
```

注意到, 如果缓冲器大小为 $M$ 个浮点数, 则必须有

$$2nl+n\leqslant M. \tag{1.4.1}$$

令 $\Gamma_1$ 是缓冲与主内存之间 (任一方向) 流动的浮点数. $\boldsymbol{B}$ 的每个元素都要取到缓冲一次, $\boldsymbol{C}$ 的每个元素都要取到缓冲一次且存回主内存一次, 还有 $\boldsymbol{A}$ 的每个元素都要取到缓冲 $N=n/l$ 次. 所以

$$\Gamma_1=3n^2+\frac{n^3}{l}.$$

为了使数据流动尽可能少, 我们在 (1.4.1) 的条件下选取尽可能大的 $l$. 于是我们取

$$l\approx\frac{1}{2}\left(\frac{M}{n}-1\right),$$

得到

$$\Gamma_1\approx 3n^2+\frac{2n^4}{M-n}.$$

(用"$\approx$", 强调我们的分析是非精确的) 如果缓冲大到可存在整个 $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 以及 $\boldsymbol{A}$ 的一列, 则 $l=n$, $\Gamma_1=4n^2$. 在另一个极端情况, 缓冲只能存三个列向量, 则 $l=1, \Gamma_1\approx n^3$.

① 如果矩阵存在磁盘中, 需要移到主内存的话, 那么下面的内容就适用了.

现在我们考虑 $\boldsymbol{A}=(\boldsymbol{A}_{\alpha\beta}), \boldsymbol{B}=(\boldsymbol{B}_{\alpha\beta})$ 和 $\boldsymbol{C}=(\boldsymbol{C}_{\alpha\beta})$ 是 $N\times N$ 分块且每块的维数都是 $l=n/N$. 在此分块下, 计算

$$\boldsymbol{C}_{\alpha\beta}=\sum_{\gamma=1}^{N}\boldsymbol{A}_{\alpha\gamma}\boldsymbol{B}_{\gamma\beta},\quad \alpha=1:N,\quad \beta=1:N,$$

可写成

**for** $\alpha=1:N$
  **for** $\beta=1:N$
    将 $C_{\alpha\beta}$ 取到缓冲
    **for** $\gamma=1:N$
      将 $A_{\alpha\gamma}$ 和 $B_{\gamma\beta}$ 取到缓冲
      $C_{\alpha\beta}=C_{\alpha\beta}+A_{\alpha\gamma}B_{\gamma\beta}$
    **end**
    将 $C_{\alpha\beta}$ 存回主内存
  **end**
**end**

在此情形主内存与缓冲之间的流量为

$$\Gamma_2=2n^2+\frac{2n^3}{l}$$

这是因为 $\boldsymbol{A}$ 的元素和 $\boldsymbol{B}$ 的元素需要读取 $N=n/l$ 次, $\boldsymbol{C}$ 的每一个元素都需读取以及存写一次. 我们能通过选取尽可能大的 $L$ 来极小化这一通量, 条件是只要缓冲中能储存下三个矩阵块, 即

$$3l^2\leqslant M.$$

令 $l\approx\sqrt{M/3}$ 即知

$$\Gamma_2\approx 2n^2+2n^3\sqrt{\frac{3}{M}}.$$

通过简单计算可知

$$\frac{\Gamma_1}{\Gamma_2}\approx\frac{3n^2+\dfrac{2n^4}{M-n}}{2n^2+2n^3\sqrt{\dfrac{3}{M}}}\geqslant\frac{3+2\dfrac{n^2}{M}}{2+2\sqrt{3}\sqrt{\dfrac{n^2}{M}}}.$$

一个重要的量是 $n^2/M$, 即矩阵大小 (浮点数) 与缓冲器大小之比. 当这个比增大时, 有

$$\frac{\Gamma_1}{\Gamma_2}\approx\frac{n}{\sqrt{3M}},$$

故知从数据进出缓冲器的角度来说, 第二种分块方案要好. 所有这些所导致的基本结论是, *分块影响数据的移动*.

### 1.4.8 分块矩阵结构

我们以讨论分块数据结构来结束本节. 允许两维数组的程序语言必须有在内存如何储存这种数组的约定. 例如, Fortran 是按列优先储存两维数组的. 这意味着一列的元素在内存中是相邻的. 所以, 如果 24 个储存单元分配给 $A \in \mathbb{R}^{4\times 6}$, 则传统的列优先储存是把矩阵元素像图 1.4.4 那样排列储存在内存之中的. 换句话说, 如果 $A \in \mathbb{R}^{m\times n}$ 存于 $v(1:mn)$, 则 $A(i,j)$ 存于 $v((j-1)m+j)$. 这种方式对于按列用到矩阵之元素的算法是有好处的, 因为列元素是连续存储的.

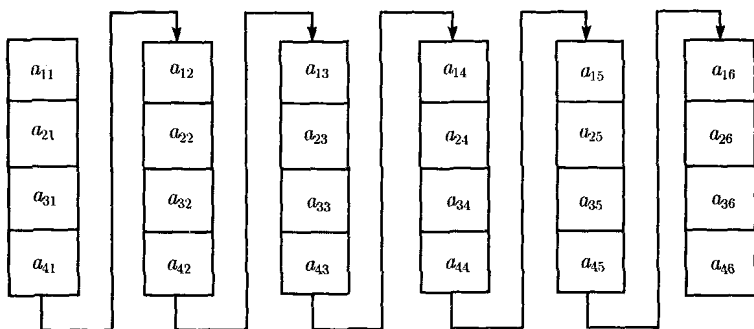



图 1.4.4 按列存储 ($4\times 6$ 例子)

在某些分块矩阵算法中, 有时把矩阵按块存比按列存更有用. 例如, 上面提到的矩阵 $A$ 可看作是一个 $2\times 3$ 分块矩阵, 每块都是 $2\times 2$ 块, 则 24 个元素在内存中可按图 1.4.5 所示排列. 这种数据结构对分块矩阵算法是很有吸引力的, 这是由于每块的元素在内存中是连续存放的.

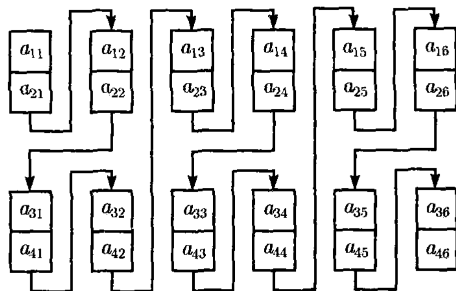



图 1.4.5 按块储存 ($4\times 6$ 例子, $2\times 2$ 子块)

## 习 题

**1.4.1** 考虑矩阵乘积 $D = ABC$, 其中 $A \in \mathbb{R}^{m\times r}$, $B \in \mathbb{R}^{r\times n}$ 和 $C \in \mathbb{R}^{n\times q}$, 假定所有矩阵都按列存储, 且执行 $k$ 维整体间的 saxpy 运算需要的时间为 $t(k) = (L+k)\mu$, 其中 $L$ 是常数, $\mu$ 是每一步的时间. 基于这一模型, 什么时候用 $D = (AB)C$ 比用 $D = A(BC)$ 来计

算 $D$ 更经济? 假定所有的矩阵乘法都用 $jki$(gaxpy) 算法.

**1.4.2** 设所有的矩阵都是按列储存, 运行长度为 $k$ 的 saxpy 需时间 $t(k) = (L+k)\mu$, 其中 $L$ 是常数, $\mu$ 是每步的时间, 问 $jki$ 形式的 saxpy 运算需要多少时间? 给出一个能有效处理 $A$ 和 $B$ 都是 $n \times n$ 上三角形矩阵的情形. 此算法的速度是否像 flop 数所表明的那样是方阵情形的 6 倍?

**1.4.3** 给出一个计算 $C = A^TBA$ 的算法, 其中 $A$ 和 $B$ 都是 $n \times n$ 矩阵具且 $B$ 是对称的. 所有内层数据都应该是整体间.

**1.4.4** 假定 $A \in \mathbb{R}^{m\times n}$ 按列存于 A.col($1:mn$). 设 $m = l_1M$ 和 $n = l_2N$, 我们把 $A$ 看成 $M \times N$ 分块矩阵. 每块大小为 $l_1 \times l_2$. 任给 $i, j, \alpha$ 和 $\beta$ 满足 $1 \leqslant i \leqslant l_1, l \leqslant j \leqslant l_2, 1 \leqslant \alpha \leqslant M$ 和 $1 \leqslant \beta \leqslant N$, 求出 $k$ 使得 A.col($k$) 储存 $A_{\alpha\beta}$ 中的 $(i, j)$ 元素, 请给出一个将 $A$ 按块储存 (见图 1.4.5) 覆盖 A.col($k$) 的算法. 请问需要多大的工作数组?

## 本节注释与参考文献

介绍向量计算的两篇很好的文章是:

J. J. Dongarra, F. G. Gustavson, and A. Karp(1984). "Implementing Linear Algebra Algorithms for Dense Matrices on a Vector Pipeline Machine," *SIAM Review* 26, 91–112.

J. M. Ortega and R. G. Voigt(1985). "Solution of Partial Differential Equations on Vector and Parallel Computers," *SIAM Review* 27, 149–240.

关于分级储存系统矩阵计算的详细讨论可见:

K. Gallivan, W. Jalby, U. Meier, and A. H. Sameh(1988). "Impact of Hierarchical Memory Systems on Linear Algebra Algorithm Design," *Int'l J. Supercomputer Applic.* 2, 12–48.

以及

W. Schönauer(1987). *Scientific Computing on Vector Computers,* North Holland, Amsterdam.

R. W. Hockney and C. R. Jesshope(1988). *Parallel Computers* 2, Adam Hilger, Bristol and Philadelphia.

以上文献讨论了各种向量处理器的模型. 关于向量计算的应用方面的论文包括:

J. J. Dongarra and A. Hinds(1979). "Unrolling Loops in Fortran," *Software Practice and Experience* 9, 219–229.

J. J. Dongarra and S. Eisenstat(1984). "Squeezing the Most Out of an Algorithm in Cray Fortran," *ACM Trans. Math. Soft.* 10, 221–230.

B. L. Buzbee(1986). "A Strategy for Vectorization," *Parallel Computing* 3, 187–192.

K. Gallivan, W. Jalby, and U. Meier(1987). "The Use of BLAS3 in Linear Algebra on a Parallel Processor with a Hierarchical Memory," *SIAM J. Sci. and Stat. Comp.* 8, 1079–1084.

J. J. Dongarra and D. Walker(1995). "Software Libraries for Linear Algebra Computations on High Performance Computers," *SIAM Review* 37, 151–180.

# 第2章　矩阵分析

矩阵计算的算法之导出和分析需要借助于线性代数的某些性质. 一些基本的性质在 2.1 节中介绍, 范数及其性质在 2.2 节和 2.3 节中介绍. 在 2.4 节中, 我们给出有限精度计算的模型并将其用于误差分析.

2.5 节和 2.6 节介绍正交性, 正交性在矩阵计算中是至关重要的. 奇异值分解和 CS 分解是两个正交约化, 它们有助于深刻理解秩以及子空间之间的距离等重要概念. 在 2.7 节我们讨论当 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 有扰动时线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 之解是如何变化的, 并给出了矩阵条件数这一重要概念.

**预备知识**

与本章内容相辅的文献包括 Forsythe and Moler(1976), Stewart(1973), Stewart and Sun(1990), 以及 Higham(1996).

## 2.1　线性代数初步

本节粗略地复习线性代数, 希望更详细了解的读者应参阅本节末给出的参考文献.

### 2.1.1　独立, 子空间, 基和维数

对于$\mathbb{R}^m$中的一组向量$\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$, 如果从 $\sum_{j=1}^n \alpha_j\boldsymbol{a}_j=0$ 可推出$\alpha(1:n)=0$, 则称 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$*线性无关*. 否则有 $\boldsymbol{a}_i$ 的非平凡组合为零, 此时称 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 是*线性相关的*.

如果 $\mathbb{R}^m$ 中的子集也是向量空间, 则称其为 $\mathbb{R}^m$ 的*子空间*. 任给一组向量 $\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\in\mathbb{R}^m$, 这组向量的所有线性组合构成一个子空间, 称为 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 的*张成空间*:

$$\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}=\left\{\sum_{j=1}^n\beta_j\boldsymbol{a}_j:\beta_j\in\mathbb{R}\right\}.$$

如果 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 是线性无关的且 $\boldsymbol{b}\in\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$, 则 $\boldsymbol{b}$ 是 $\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n$ 的唯一的线性组合.

如果 $S_1,\cdots,S_k$ 是 $\mathbb{R}^m$ 的子空间, 则它们的和是由 $S=\{\boldsymbol{a}_1+\boldsymbol{a}_2+\cdots+\boldsymbol{a}_k: \boldsymbol{a}_i\in S_i, i=1:k\}$ 所定义的子空间. 如果每个 $\boldsymbol{v}\in S$ 都有唯一的表示方式 $\boldsymbol{v}=\boldsymbol{a}_1+\cdots+\boldsymbol{a}_k, \boldsymbol{a}_i\in S_i$, 则称 $S$ 为*直和*, 在这种情形我们记为 $S=S_1\oplus\cdots\oplus S_k$. $S_i$ 的交集 $S=S_1\cap S_2\cap\cdots\cap S_k$ 也是一个子空间.

如果子集 $\{\boldsymbol{a}_{i_1},\cdots,\boldsymbol{a}_{i_k}\}$ 线性无关且不是 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 的任何线性无关子集的真子集, 则称它是 $\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 的**极大线性无关子集**. 如果 $\{\boldsymbol{a}_{i_1},\cdots,\boldsymbol{a}_{i_k}\}$ 是极大线性无关子集, 则 $\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}=\mathrm{span}\{\boldsymbol{a}_{i_1},\cdots,\boldsymbol{a}_{i_k}\}$ 而且 $\{\boldsymbol{a}_{i_1},\cdots,\boldsymbol{a}_{i_k}\}$ 是 $\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}$ 的一组基. 如果 $S\subseteq\mathbb{R}^m$ 是一子空间, 则可找到线性无关的基向量 $\boldsymbol{a}_1,\cdots,\boldsymbol{a}_k\in S$, 满足 $S=\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_k\}$. 一个子空间 $S$ 的所有基都有同样多的元素. 这个数目是 $S$ 的**维数**, 记为 $\dim(S)$.

### 2.1.2 域, 零空间和秩

关于 $m\times n$ 矩阵 $\boldsymbol{A}$ 有两个重要的子空间. $\boldsymbol{A}$ 的**值域**定义为

$$\mathrm{ran}(\boldsymbol{A})=\{\boldsymbol{y}\in\mathbb{R}^m:\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x},\quad \text{对某一}\boldsymbol{x}\in\mathbb{R}^n\}.$$

$\boldsymbol{A}$ 的**零空间**定义为

$$\mathrm{null}(\boldsymbol{A})=\{\boldsymbol{x}\in\mathbb{R}^n:\boldsymbol{A}\boldsymbol{x}=\boldsymbol{0}\}.$$

如果 $\boldsymbol{A}=[\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n]$ 是一个列划分, 则

$$\mathrm{ran}(\boldsymbol{A})=\mathrm{span}\{\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n\}.$$

矩阵 $\boldsymbol{A}$ 的**秩**定义为

$$\mathrm{rank}(\boldsymbol{A})=\dim(\mathrm{ran}(\boldsymbol{A})).$$

可以证明 $\mathrm{rank}(\boldsymbol{A})=\mathrm{rank}(\boldsymbol{A}^{\mathrm{T}})$. 如果 $\mathrm{rank}(\boldsymbol{A})<\min\{m,n\}$, 我们说 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 是**秩亏损的**. 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 则

$$\dim(\mathrm{null}(\boldsymbol{A}))+\mathrm{rank}(\boldsymbol{A})=n.$$

### 2.1.3 逆矩阵

$n\times n$ 单位矩阵 $\boldsymbol{I}_n$ 是由列划分

$$\boldsymbol{I}_n=[\boldsymbol{e}_1,\cdots,\boldsymbol{e}_n]$$

所定义, 其中 $\boldsymbol{e}_k$ 是第 $k$ 个"典范"向量

$$\boldsymbol{e}_k=(\underbrace{0,\cdots,0}_{k-1},1,\underbrace{0,\cdots,0}_{n-k})^{\mathrm{T}}.$$

坐标向量在矩阵分析中常常用到, 当维数不明显时, 我们用上标记号, 即 $\boldsymbol{e}_k^{(n)}\in\mathbb{R}^n$,

如果 $\boldsymbol{A}$ 和 $\boldsymbol{X}$ 属于 $\mathbb{R}^{n\times n}$ 且满足 $\boldsymbol{A}\boldsymbol{X}=\boldsymbol{I}$, 则 $\boldsymbol{X}$ 是 $\boldsymbol{A}$ 的**逆矩阵**, 被记为 $\boldsymbol{A}^{-1}$. 如果 $\boldsymbol{A}^{-1}$ 存在, 则称 $\boldsymbol{A}$ 为非奇异的, 否则我们说 $\boldsymbol{A}$ 是**奇异的**.

逆矩阵的一些性质在矩阵计算中起重要作用. 乘积的逆等于逆的逆序积:

$$(\boldsymbol{A}\boldsymbol{B})^{-1}=\boldsymbol{B}^{-1}\boldsymbol{A}^{-1}. \tag{2.1.1}$$

逆的转置等于转置的逆:

$$(\boldsymbol{A}^{-1})^{\mathrm{T}}=(\boldsymbol{A}^{\mathrm{T}})^{-1}\equiv\boldsymbol{A}^{-\mathrm{T}}. \tag{2.1.2}$$

恒等式

$$B^{-1} = A^{-1} - B^{-1}(B - A)A^{-1} \tag{2.1.3}$$

表明逆的变化与矩阵变化的关系.

Sherman-Morrison-Woodbury 公式给出了 $(A + UV^{\mathrm{T}})$ 之逆的一个方便的表达式:

$$(A + UV^{\mathrm{T}})^{-1} = A^{-1} - A^{-1}U(I + V^{\mathrm{T}}A^{-1}U)^{-1}V^{\mathrm{T}}A^{-1}, \tag{2.1.4}$$

其中 $A \in \mathbb{R}^{n\times n}$, $U$ 和 $V \in \mathbb{R}^{n\times k}$. 矩阵的秩 $k$ 变化导致逆矩阵秩 $k$ 的变化. 在 (2.1.4) 中我们假定 $A$ 和 $I + V^{\mathrm{T}}A^{-1}U$ 都是非奇异的.

上述事实可以通过直接验算逆矩阵应满足的条件来证明. 例如, 以下是 (2.1.3) 的证明:

$$B(A^{-1} - B^{-1}(B - A)A^{-1}) = BA^{-1} - (B - A)A^{-1} = I.$$

### 2.1.4 行列式

设 $A = (a) \in \mathbb{R}^{1\times 1}$, 则它的行列式是 $\det(A) = a$. $A \in \mathbb{R}^{n\times n}$ 的行列式可由 $n-1$ 阶行列式来定义:

$$\det(A) = \sum_{j=1}^{n}(-1)^{j+1}a_{1j}\det(A_{1j}).$$

这里 $A_{1j}$ 是从 $A$ 中删去第一行和第 $j$ 列后所得到的 $(n-1)\times(n-1)$ 矩阵. 行列式的有用性质包括

$$\begin{aligned}
&\det(AB) = \det(A)\det(B), && A, B \in \mathbb{R}^{n\times n},\\
&\det(A^{\mathrm{T}}) = \det(A), && A \in \mathbb{R}^{n\times n},\\
&\det(cA) = c^n\det(A), && c \in \mathbb{R}, A \in \mathbb{R}^{n\times n},\\
&\det(A) \neq 0 \Leftrightarrow A\text{是非奇异}, && A \in \mathbb{R}^{n\times n}.
\end{aligned}$$

### 2.1.5 微分

设 $\alpha$ 是标量且 $A(\alpha)$ 是由元素 $a_{ij}(\alpha)$ 组成的 $m \times n$ 矩阵, 如果对一切 $i$ 和 $j$, $a_{ij}(\alpha)$ 都是可微函数, 则我们记 $\dot{A}\,(\alpha)$ 为矩阵

$$\dot{A}\,(\alpha) = \frac{\mathrm{d}}{\mathrm{d}\alpha}A(\alpha) = \left(\frac{\mathrm{d}}{\mathrm{d}\alpha}a_{ij}(\alpha)\right) = (\dot{a}_{ij}(\alpha)).$$

参数化矩阵的微分对研究许多矩阵问题的敏感度是很方便的.

## 习 题

**2.1.1** 证明: 如果 $A \in \mathbb{R}^{m\times n}$ 的秩为 $p$, 则存在 $X \in \mathbb{R}^{m\times p}$ 和 $Y \in \mathbb{R}^{n\times p}$ 使得 $A = XY^{\mathrm{T}}$, 其中 $\mathrm{rank}(X) = \mathrm{rank}(Y) = p$.

**2.1.2** 假定 $A(\alpha) \in \mathbb{R}^{m\times r}$ 和 $B(\alpha) \in \mathbb{R}^{r\times n}$ 的元素都是标量 $\alpha$ 的可微函数, 证明

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}[\boldsymbol{A}(\alpha)\boldsymbol{B}(\alpha)] = \left[\frac{\mathrm{d}}{\mathrm{d}\alpha}\boldsymbol{A}(\alpha)\right]\boldsymbol{B}(\alpha) + \boldsymbol{A}(\alpha)\left[\frac{\mathrm{d}}{\mathrm{d}\alpha}\boldsymbol{B}(\alpha)\right].$$

**2.1.3** 假定 $\boldsymbol{A}(\alpha) \in \mathbb{R}^{n\times n}$ 的元素都是标量 $\alpha$ 的可微函数. 设 $\boldsymbol{A}(\alpha)$ 对一切 $\alpha$ 都可逆, 证明

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}[\boldsymbol{A}(\alpha)^{-1}] = -\boldsymbol{A}(\alpha)^{-1}\left[\frac{\mathrm{d}}{\mathrm{d}\alpha}\boldsymbol{A}(\alpha)\right]\boldsymbol{A}(\alpha)^{-1}.$$

**2.1.4** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{b} \in \mathbb{R}^n$ 且 $\phi(x) = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{b}$. 证明 $\phi$ 的梯度是 $\nabla\phi(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{A}^{\mathrm{T}} + \boldsymbol{A})\boldsymbol{x} - \boldsymbol{b}$.

**2.1.5** 假定 $\boldsymbol{A}$ 和 $\boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}$ 都非奇异, 其中 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 且 $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$. 证明: 如果 $\boldsymbol{x}$ 是 $(\boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}})\boldsymbol{x} = \boldsymbol{b}$ 的解, 则它也是带扰动右端项问题 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} + \alpha\boldsymbol{u}$ 的解. 试用 $\boldsymbol{A}, \boldsymbol{u}$ 和 $\boldsymbol{v}$ 表达 $\alpha$.

### 本节注释与参考文献

有许多线性代数入门教材. 在它们之中, 下面这些是特别有用的.

P. R. Halmos(1958). *Finite Dimensional Vector Spaces,* 2nd ed., Van Nostrand-Reinhold, Princeton.

S. J. Leon(1980). *Linear Algebra with Applications.* Macmillan, New York.

G. Strang(1993). *Introduction to Linear Algebra*, Wellesley-Cambridge Press, Wellesley MA.

D. Lay(1994). *Linear Algebra and Its Applications*, Addison-Wesley, Reading, MA.

C. Meyer(1997). *A Course in Applied Linear Algebra*, SIAM Publications, Philadelphia, PA.

更多的高级教程包括 Gantmacher(1959), Horn and Johnson(1985, 1991) 以及

A. S. Householder(1964). *The Theory of Matrices in Numerical Analysis*, Ginn(Blaisdell), Boston.

M. Marcus and H. Minc(1964.) *A Survey of Matrix Theory and Matrix Inequalities*, Allyn and Bacon, Boston.

J. N. Franklin(1968). *Matrix Theory*, Prentice Hall, Englewood Cliffs, NJ.

R. Bellman(1970). *Introduction to Matrix Analysis, Second Edition*, McGraw-Hill, New York.

P. Lancaster and M. Tismenetsky(1985). *The Theory of Matrices, Second Edition*, Academic Press, New York.

J. M. Ortega(1987). *Matrix Theory:A Second Course*, Plenum Press, New York.

## 2.2 向量范数

范数对向量空间的作用就像实数轴上的绝对值: 它提供距离的一个度量. 更确切地说, $\mathbb{R}^n$ 与其上的一个范数定义了一个度量空间. 从而, 我们在研究向量及向量值函数时有邻域、开集、收敛和连续性等熟悉的概念.

### 2.2.1 定义

在 $\mathbb{R}^n$ 上的向量范数是满足如下性质的从 $\mathbb{R}^n$ 到 $\mathbb{R}$ 的函数:

$$
\begin{aligned}
&f(\boldsymbol{x}) \geqslant 0, && \boldsymbol{x} \in \mathbb{R}^n (f(\boldsymbol{x}) = 0 \text{当且仅当} \boldsymbol{x} = \boldsymbol{0});\\
&f(\boldsymbol{x}+\boldsymbol{y}) \leqslant f(\boldsymbol{x}) + f(\boldsymbol{y}), && \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n;\\
&f(\alpha \boldsymbol{x}) = |\alpha| f(\boldsymbol{x}), && \alpha \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^n.
\end{aligned}
$$

我们用双竖线记号来表示此函数：$f(x) = \|\boldsymbol{x}\|$. 双竖线的下标用来区分不同的范数.

一类有用的向量范数是 $p$范数, 其定义为

$$
\|\boldsymbol{x}\|_p = (|x_1|^p + \cdots + |x_n|^p)^{\frac{1}{p}}, \quad p \geqslant 1. \tag{2.2.1}
$$

其中最重要的是 1 范数, 2 范数和无穷范数

$$
\begin{aligned}
&\|\boldsymbol{x}\|_1 = |x_1| + \cdots + |x_n|,\\
&\|\boldsymbol{x}\|_2 = (|x_1|^2 + \cdots + |x_n|^2)^{\frac{1}{2}} = (\boldsymbol{x}^{\mathrm{T}} \boldsymbol{x})^{\frac{1}{2}},\\
&\|\boldsymbol{x}\|_\infty = \max_{1 \leqslant i \leqslant n} |x_i|.
\end{aligned}
$$

在范数 $\|\cdot\|$ 意义下的单位向量是指满足 $\|\boldsymbol{x}\| = 1$ 的向量 $\boldsymbol{x}$.

### 2.2.2 向量范数性质

关于 $p$ 范数的一个经典结果是 Hölder不等式

$$
|\boldsymbol{x}^{\mathrm{T}} \boldsymbol{y}| \leqslant \|\boldsymbol{x}\|_p \|\boldsymbol{y}\|_q, \quad \frac{1}{p} + \frac{1}{q} = 1. \tag{2.2.2}
$$

它的一个非常重要的特殊情形是 Cauchy-Schwartz不等式

$$
|\boldsymbol{x}^{\mathrm{T}} \boldsymbol{y}| \leqslant \|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2. \tag{2.2.3}
$$

$\mathbb{R}^n$ 上的所有范数都是等价的，也就是说, 如果 $\|\cdot\|_\alpha$ 和 $\|\cdot\|_\beta$ 是 $\mathbb{R}^n$ 上的范数, 则存在正常数 $c_1$ 和 $c_2$ 使得

$$
c_1 \|\boldsymbol{x}\|_\alpha \leqslant \|\boldsymbol{x}\|_\beta \leqslant c_2 \|\boldsymbol{x}\|_\alpha \tag{2.2.4}
$$

对一切 $\boldsymbol{x} \in \mathbb{R}^n$ 都成立. 例如, 如果 $\boldsymbol{x} \in \mathbb{R}^n$, 则有

$$
\|\boldsymbol{x}\|_2 \leqslant \|\boldsymbol{x}\|_1 \leqslant \sqrt{n} \|\boldsymbol{x}\|_2, \tag{2.2.5}
$$

$$
\|\boldsymbol{x}\|_\infty \leqslant \|\boldsymbol{x}\|_2 \leqslant \sqrt{n} \|\boldsymbol{x}\|_\infty, \tag{2.2.6}
$$

$$
\|\boldsymbol{x}\|_\infty \leqslant \|\boldsymbol{x}\|_1 \leqslant n \|\boldsymbol{x}\|_\infty. \tag{2.2.7}
$$

### 2.2.3 绝对误差和相对误差

假设 $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ 是 $\boldsymbol{x} \in \mathbb{R}^n$ 的一个近似. 对给定向量范数 $\|\cdot\|$, 我们称

$$
\varepsilon_{abs} = \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|
$$

为 $\hat{\boldsymbol{x}}$ 的绝对误差. 如果 $\boldsymbol{x} \neq \boldsymbol{0}$, 则称

$$
\varepsilon_{rel} = \frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|}{\|\boldsymbol{x}\|}
$$

为 $\hat{x}$ 的相对误差. $\infty$ 范数意义下的相对误差可以换成 $\hat{x}$ 具有正确的有效位数字的说法. 例如, 如果

$$\frac{\|\hat{x}-x\|_\infty}{\|x\|_\infty} \approx 10^{-p},$$

则 $\hat{x}$ 的最大分量至少有大约 $p$ 位正确的有效数字.

**例 2.2.1** 如果 $x = (1.234, 0.056\ 74)^{\mathrm{T}}$, $\hat{x} = (1.235, 0.051\ 28)^{\mathrm{T}}$, 则 $\|\hat{x}-x\|_\infty/\|x\|_\infty \approx 0.0043 \approx 10^{-3}$. 注意, $\hat{x}_1$ 有大约 3 位有效数字是正确的, 而 $\hat{x}_2$ 仅有一位正确的有效数字.

### 2.2.4 收敛性

如果

$$\lim_{k\to\infty} \|x^{(k)} - x\| = 0,$$

则称 $n$ 维向量序列 $\{x^{(k)}\}$ 是收敛的. 从 (2.2.4) 可知, 由在 $\alpha$ 范数意义下收敛可推出在 $\beta$ 范数意义下收敛, 反之亦然.

## 习 题

**2.2.1** 设 $x \in \mathbb{R}^n$, 证明 $\lim\limits_{p\to\infty} \|x\|_p = \|x\|_\infty$.

**2.2.2** 利用不等式 $0 \leqslant (ax+by)^{\mathrm{T}}(ax+by)$ 并且选取适当的 $a$ 和 $b$, 证明 Cauchy-Schwartz 不等式 (2.2.3).

**2.2.3** 验证 $\|\cdot\|_1$, $\|\cdot\|_2$ 和 $\|\cdot\|_\infty$ 都是向量范数.

**2.2.4** 证明 (2.2.5)~(2.2.7), 问何时这些等式成立?

**2.2.5** 证明在 $\mathbb{R}^n$ 中 $x^{(i)} \to x$ 当且仅当对 $k=1:n$ 有 $x_k^{(i)} \to x_k$.

**2.2.6** 通过验证不等式 $|\ \|x\| - \|y\|\ | \leqslant \|x-y\|$ 来证明 $\mathbb{R}^n$ 中任何向量范数是一致连续的.

**2.2.7** 设 $\|\cdot\|$ 是 $\mathbb{R}^m$ 上的向量范数且 $A \in \mathbb{R}^{m\times n}$. 证明如果 $\mathrm{rank}(A) = n$, 则 $\|x\|_A = \|Ax\|$ 是 $\mathbb{R}^n$ 上的向量范数.

**2.2.8** 设 $x, y \in \mathbb{R}^n$, 定义 $\mathbb{R}\to\mathbb{R}$ 的函数 $\psi(\alpha) = \|x-\alpha y\|_2$. 证明 $\psi$ 在 $\alpha = x^{\mathrm{T}}y/y^{\mathrm{T}}y$ 时达到极小.

**2.2.9** (a) 证明 $\|x\|_p = (|x_1|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$ 是 $\mathbb{C}^n$ 上的向量范数. (b) 证明: 如果 $x \in \mathbb{C}^n$ 则 $\|x\|_p \leqslant c(\|\mathrm{Re}(x)\|_p + \|\mathrm{Im}(x)\|_p)$. (c) 找出常数 $c_n$ 使得 $c_n(\|\mathrm{Re}(x)\|_p + \|\mathrm{Im}(x)\|_p) \leqslant \|x\|_2$ 对一切 $x \in \mathbb{C}^n$ 都成立.

**2.2.10** 证明或举例反证

$$v \in \mathbb{R}^n \Rightarrow \|v\|_1\|v\|_\infty \leqslant \frac{1+\sqrt{n}}{2}\|v\|_2.$$

## 本节注释与参考文献

虽然向量范数“仅仅”是绝对值概念的推广, 有些微妙之处是值得注意的.

J. D. Pryce(1984). "A New Measure of Relative Error for Vectors", *SIAM J. Num. Anal.* *21*, 202–221.

## 2.3 矩阵范数

分析矩阵算法时常需要利用矩阵范数. 例如, 线性方程组的解法在系数矩阵"几乎奇异"时效果可能很差, 为了量化"几乎奇异"这一概念我们需要矩阵空间之距离的度量. 矩阵范数提供了此度量.

### 2.3.1 定义

因为 $\mathbb{R}^{m\times n}$ 与 $\mathbb{R}^{mn}$ 是同构的, 矩阵范数的定义应等价于向量范数的定义. 特别地, $f:\mathbb{R}^{m\times n}\to\mathbb{R}$ 在满足

$$\begin{aligned}
&f(\boldsymbol{A})\geqslant 0, && \boldsymbol{A}\in\mathbb{R}^{m\times n},\quad (f(\boldsymbol{A})=0\text{当且仅当}\boldsymbol{A}=\boldsymbol{0}),\\
&f(\boldsymbol{A}+\boldsymbol{B})\leqslant f(\boldsymbol{A})+f(\boldsymbol{B}), && \boldsymbol{A},\boldsymbol{B}\in\mathbb{R}^{m\times n},\\
&f(\alpha\boldsymbol{A})=|\alpha|f(\boldsymbol{A}), && \alpha\in\mathbb{R},\boldsymbol{A}\in\mathbb{R}^{m\times n},
\end{aligned}$$

这三个条件时是矩阵范数. 与向量范数一样, 我们用带下标双竖线来表示矩阵范数, 即 $\|\boldsymbol{A}\|=f(\boldsymbol{A})$.

在数值线性代数中, 最常用的矩阵范数是 $F$ 范数 (Frobenius 范数)

$$\|\boldsymbol{A}\|_F=\sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} \tag{2.3.1}$$

和 $p$ 范数

$$\|\boldsymbol{A}\|_p=\sup_{x\neq 0}\frac{\|\boldsymbol{Ax}\|_p}{\|\boldsymbol{x}\|_p}. \tag{2.3.2}$$

注意到矩阵 $p$ 范数的定义是基于上一节所讨论的向量 $p$ 范数, 验证 (2.3.1) 和 (2.3.2) 是矩阵范数将作为一练习. 很明显, $\|\boldsymbol{A}\|_p$ 是将 $\boldsymbol{A}$ 作用到 $p$ 范数单位向量所得到的最大向量的 $p$ 范数:

$$\|\boldsymbol{A}\|_p=\sup_{x\neq 0}\left\|\boldsymbol{A}\left(\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_p}\right)\right\|_p=\max_{\|x\|_p=1}\|\boldsymbol{Ax}\|_p.$$

重要的是要认识到 (2.3.1) 和 (2.3.2) 定义了范数簇, 例如 $\mathbb{R}^{3\times 2}$ 上的 2 范数与 $\mathbb{R}^{5\times 6}$ 上的 2 范数是不同的函数, 因而, 很容易验证的如下不等式

$$\|\boldsymbol{AB}\|_p\leqslant\|\boldsymbol{A}\|_p\|\boldsymbol{B}\|_p,\quad \boldsymbol{A}\in\mathbb{R}^{m\times n},\quad \boldsymbol{B}\in\mathbb{R}^{n\times q} \tag{2.3.3}$$

实质上是关于三种不同范数之间关系的结论. 规范地说, 如果对所有 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 和 $\boldsymbol{B}\in\mathbb{R}^{n\times q}$ 有 $f_1(\boldsymbol{AB})\leqslant f_2(\boldsymbol{A})f_3(\boldsymbol{B})$, 则称 $\mathbb{R}^{m\times q}$, $\mathbb{R}^{m\times n}$ 和 $\mathbb{R}^{n\times q}$ 上的范数 $f_1,f_2$ 和 $f_3$ 是*相互相容的*.

并非所有矩阵范数都满足可乘性质：

$$\|\boldsymbol{AB}\| \leqslant \|\boldsymbol{A}\|\|\boldsymbol{B}\|. \tag{2.3.4}$$

例如，设 $\|\boldsymbol{A}\|_\Delta = \max|a_{ij}|$ 和

$$\boldsymbol{A} = \boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

则 $\|\boldsymbol{AB}\|_\Delta > \|\boldsymbol{A}\|_\Delta\|\boldsymbol{B}\|_\Delta$. 大多数情形下，我们研究的范数满足 (2.3.4).

$p$ 范数有一重要性质，即对任何 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 及 $\boldsymbol{x} \in \mathbb{R}^n$ 有 $\|\boldsymbol{Ax}\|_p \leqslant \|\boldsymbol{A}\|_p\|\boldsymbol{x}\|_p$. 更一般地，对 $\mathbb{R}^n$ 上的任意向量范数 $\|\cdot\|_\alpha$ 和 $\mathbb{R}^m$ 上的任意向量范数 $\|\cdot\|_\beta$ 有 $\|\boldsymbol{Ax}\|_\beta \leqslant \|\boldsymbol{A}\|_{\alpha,\beta}\cdot\|\boldsymbol{x}\|_\alpha$，其中 $\|\boldsymbol{A}\|_{\alpha,\beta}$ 是由

$$\|\boldsymbol{A}\|_{\alpha,\beta} = \sup_{x\neq 0}\frac{\|\boldsymbol{Ax}\|_\beta}{\|\boldsymbol{x}\|_\alpha} \tag{2.3.5}$$

所定义的矩阵范数. 我们称 $\|\cdot\|_{\alpha,\beta}$ 是从属于向量范数 $\|\cdot\|_\alpha$ 和 $\|\cdot\|_\beta$ 的. 由于集合 $\{\boldsymbol{x} \in \mathbb{R}^n : \|x\|_\alpha = 1\}$ 是紧的，而且 $\|\cdot\|_\beta$ 是连续的，故有

$$\|\boldsymbol{A}\|_{\alpha,\beta} = \max_{\|x\|_\alpha=1}\|\boldsymbol{Ax}\|_\beta = \|\boldsymbol{Ax}^*\|_\beta, \tag{2.3.6}$$

对某一 $\alpha$ 范数单位向量 $\boldsymbol{x}^* \in \mathbb{R}^n$ 成立.

### 2.3.2 矩阵范数的一些性质

Frobenius 范数和 $p$ 范数 (特别是 $p = 1, 2, \infty$) 满足一些在矩阵计算的分析中常常用到的不等式. 对 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$，我们有

$$\|\boldsymbol{A}\|_2 \leqslant \|\boldsymbol{A}\|_F \leqslant \sqrt{n}\|\boldsymbol{A}\|_2, \tag{2.3.7}$$

$$\max_{i,j}|a_{ij}| \leqslant \|\boldsymbol{A}\|_2 \leqslant \sqrt{mn}\max_{i,j}|a_{ij}|, \tag{2.3.8}$$

$$\|\boldsymbol{A}\|_1 = \max_{1\leqslant j\leqslant n}\sum_{i=1}^m|a_{ij}|, \tag{2.3.9}$$

$$\|\boldsymbol{A}\|_\infty = \max_{1\leqslant i\leqslant m}\sum_{j=1}^n|a_{ij}|, \tag{2.3.10}$$

$$\frac{1}{\sqrt{n}}\|\boldsymbol{A}\|_\infty \leqslant \|\boldsymbol{A}\|_2 \leqslant \sqrt{m}\|\boldsymbol{A}\|_\infty, \tag{2.3.11}$$

$$\frac{1}{\sqrt{m}}\|\boldsymbol{A}\|_1 \leqslant \|\boldsymbol{A}\|_2 \leqslant \sqrt{n}\|\boldsymbol{A}\|_1. \tag{2.3.12}$$

设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, 1 \leqslant i_1 \leqslant i_2 \leqslant m$ 以及 $1 \leqslant j_1 \leqslant j_2 \leqslant n$，则

$$\|A(i_1 : i_2, j_1 : j_2)\|_p \leqslant \|\boldsymbol{A}\|_p. \tag{2.3.13}$$

这些关系式的证明不难，均留作习题.

如果 $\lim\limits_{k\to\infty}\|\boldsymbol{A}^{(k)}-\boldsymbol{A}\|=0$, 则称序列 $\{\boldsymbol{A}^{(k)}\}\in\mathbb{R}^{m\times n}$ 是收敛的. 范数的选取是无关紧要, 这是因为 $\mathbb{R}^{m\times n}$ 上的所有范数都是等价的.

### 2.3.3 矩阵 2 范数

矩阵 1 范数和 $\infty$ 范数的一个良好性质是它们可从 (2.3.9) 和 (2.3.10) 容易地计算. 2 范数的特征要复杂得多.

**定理 2.3.1** 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 则存在一个 $n$ 维 2 范数单位向量 $\boldsymbol{z}$, 使得 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z}=\mu^2\boldsymbol{z}$, 其中 $\mu=\|\boldsymbol{A}\|_2$.

**证明** 假定 $\boldsymbol{z}\in\mathbb{R}^n$ 是使得 $\|\boldsymbol{A}\boldsymbol{z}\|_2=\|\boldsymbol{A}\|_2$ 的单位向量. 由于 $\boldsymbol{z}$ 极大化函数

$$g(\boldsymbol{x})=\frac{1}{2}\frac{\|\boldsymbol{A}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2}=\frac{1}{2}\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}},$$

故有 $\nabla g(\boldsymbol{z})=0$, 其中 $\nabla_g$ 是 $g$ 的梯度. 详细的微分计算表明, 对一切 $i=1:n$, 有

$$\frac{\partial g(z)}{\partial z_i}=\left[(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z})\sum_{j=1}^{n}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})_{ij}\boldsymbol{z}_j-(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z})z_i\right]\Big/(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z})^2.$$

利用向量记号和 $\nabla g(z)=0$, 由上式得到 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z}=(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z})\boldsymbol{z}$. 令 $\mu=\|\boldsymbol{A}z\|_2$ 即知定理成立. □

该定理表明 $\|\boldsymbol{A}\|_2^2$ 是多项式 $p(\lambda)=\det(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}-\lambda z)$ 的零点. 精确地说, $\boldsymbol{A}$ 的 2 范数是 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的最大特征值之平方根. 在第 7 章和第 8 章我们将进一步讨论特征值. 现在, 我们仅注意 2 范数的计算是需要迭代而且肯定比矩阵 1 范数或 $\infty$ 范数之计算要复杂. 幸运的是, 如果目的是得到 $\|\boldsymbol{A}\|_2$ 的数量级的估计, 则可利用 (2.3.7), (2.3.11) 或 (2.3.12).

作为“范数分析”的例子, 下面是 2 范数估计的一个巧妙结果.

**推论 2.3.2** 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 则 $\|\boldsymbol{A}\|_2\leqslant\sqrt{\|\boldsymbol{A}\|_1\|\boldsymbol{A}\|_\infty}$.

**证明** 设 $\boldsymbol{z}\neq\boldsymbol{0}$ 满足 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z}=\mu^2\boldsymbol{z}$, 其中 $\mu=\|\boldsymbol{A}\|_2$. 则有

$$\mu^2\|\boldsymbol{z}\|_1=\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z}\|_1\leqslant\|\boldsymbol{A}^{\mathrm{T}}\|_1\|\boldsymbol{A}\|_1\|\boldsymbol{z}\|_1=\|\boldsymbol{A}\|_\infty\|\boldsymbol{A}\|_1\|\boldsymbol{z}\|_1.$$ □

### 2.3.4 扰动与逆矩阵

我们常用范数来量化扰动的影响或者证明一个矩阵序列收敛于一特定的极限. 作为范数的这些运用之一例子, 我们把 $\boldsymbol{A}^{-1}$ 的变化表示为 $\boldsymbol{A}$ 的变化之函数.

**引理 2.3.3** 如果 $\boldsymbol{F}\in\mathbb{R}^{n\times n}$ 且 $\|\boldsymbol{F}\|_p<1$, 则 $\boldsymbol{I}-\boldsymbol{F}$ 非奇异且

$$(\boldsymbol{I}-\boldsymbol{F})^{-1}=\sum_{k=0}^{\infty}\boldsymbol{F}^k,$$

而且

$$\|(\boldsymbol{I}-\boldsymbol{F})^{-1}\|_p\leqslant\frac{1}{1-\|\boldsymbol{F}\|_p}.$$

**证明** 假定 $I-F$ 奇异. 则对某非零 $x$ 有 $(I-F)x=0$. 从而由 $\|x\|_p=\|Fx\|_p$ 可推出 $\|F\|_p\geqslant 1$, 矛盾. 所以 $I-F$ 非奇异. 为得到逆矩阵之表达式, 考虑恒等式

$$\left(\sum_{k=0}^{N} F^k\right)(I-F)=I-F^{N+1}.$$

由于 $\|F\|_p<1$, 从 $\|F^{N+1}\|_p\leqslant\|F\|_p^{N+1}$ 可知 $\lim\limits_{N\to\infty}F^{N+1}=0$.

于是

$$\left(\lim_{N\to\infty}\sum_{k=0}^{N} F^k\right)(I-F)=I.$$

故知 $(I-F)^{-1}=\lim\limits_{N\to\infty}\sum\limits_{k=0}^{N}F^k$. 由此不难证明

$$\|(I-F)^{-1}\|_p\leqslant\sum_{k=0}^{\infty}\|F\|_p^k=\frac{1}{1-\|F\|_p}. \qquad \square$$

注意到该引理的一个推论是 $\|(I-F)^{-1}-I\|_p\leqslant\|F\|_p/(I-\|F\|_p)$. 于是, 当 $\varepsilon\ll 1$ 时, $I$ 中的 $O(\varepsilon)$ 扰动将导致其逆的 $O(\varepsilon)$ 扰动. 我们把这一结果推广到一般矩阵.

**定理 2.3.4** 设 $A$ 非奇异且 $r=\|A^{-1}E\|_p<1$, 则 $A+E$ 非奇异且 $\|(A+E)^{-1}-A^{-1}\|_p\leqslant\|E\|_p\|A^{-1}\|_p^2/(1-r)$.

**证明** 由于 $A$ 非奇异, $A+E=A(I-F)$, 其中 $F=-A^{-1}E$. 因为 $\|F\|_p=r<1$, 从引理 2.3.3 知 $I-F$ 非奇异且 $\|(I-F)^{-1}\|_p<1/(1-r)$. 而 $(A+E)^{-1}=(I-F)^{-1}A^{-1}$, 所以

$$\|(A+E)^{-1}\|_p\leqslant\frac{\|A^{-1}\|_p}{1-r}.$$

等式 (2.1.3) 表明 $(A+E)^{-1}-A^{-1}=-A^{-1}E(A+E)^{-1}$. 因此两边取范数得

$$\begin{aligned}\|(A+E)^{-1}-A^{-1}\|_p&\leqslant\|A^{-1}\|_p\|E\|_p\|(A+E)^{-1}\|_p\\&\leqslant\frac{\|A^{-1}\|_p^2\|E\|_p}{1-r}.\end{aligned} \qquad \square$$

## 习　题

**2.3.1** 证明 $\|AB\|_p\leqslant\|A\|_p\|B\|_p$, 其中 $1\leqslant p\leqslant\infty$.

**2.3.2** 设 $B$ 是 $A$ 的任何子矩阵, 证明 $\|B\|_p\leqslant\|A\|_p$.

**2.3.3** 证明: 如果 $D=\mathrm{diag}(\mu_1,\cdots,\mu_k)\in\mathbb{R}^{m\times n}$, $k=\min\{m,n\}$, 则 $\|D\|_p=\max|\mu_i|$

**2.3.4** 证明 (2.3.7) 和 (2.3.8).

**2.3.5** 证明 (2.3.9) 和 (2.3.10).

**2.3.6** 证明 (2.3.11) 和 (2.3.12).

**2.3.7** 证明 (2.3.13).

**2.3.8** 证明: 如果 $0 \neq s \in \mathbb{R}^n$, $E \in \mathbb{R}^{n\times n}$, 则

$$\left\| E\left(I - \frac{ss^{\mathrm{T}}}{s^{\mathrm{T}}s}\right)\right\|_F^2 = \|E\|_F^2 - \frac{\|Es\|_2^2}{s^{\mathrm{T}}s}.$$

**2.3.9** 设 $u \in \mathbb{R}^m$ 和 $v \in \mathbb{R}^n$. 证明: 如果 $E = uv^{\mathrm{T}}$ 则

$$\|E\|_F = \|E\|_2 = \|u\|_2\|v\|_2, \quad \|E\|_\infty \leqslant \|u\|_\infty\|v\|_1.$$

**2.3.10** 设 $A \in \mathbb{R}^{m\times n}$, $y \in \mathbb{R}^m$ 以及 $0 \neq s \in \mathbb{R}^n$. 证明 $E = (y - As)s^{\mathrm{T}}/s^{\mathrm{T}}s$ 是所有满足 $(A+E)s = y$ 的具有最小 2 范数的 $m \times n$ 矩阵.

### 本节注释与参考文献

关于矩阵范数及向量范数的深入讨论, 可见:


F. L. Bauer and C. T. Fike(1960). "Norms and Exclusion Theorems," *Numer. Math.* 2, 137–144.

L. Mirsky(1960). "Symmetric Gauge Functions and Unitarily Invariant Norms," *Quart. J. Math.* 11, 50–59.

A. S. Householder(1964). *The Theory of Matrices in Numerical Analysis*, Dover Publications, New York.

N. J. Higham(1992). "Estimating the Matrix p-Norm," *Numer. Math.* 62, 539–556.


## 2.4 有限精度矩阵计算

从某种程度上数, 正是舍入误差使得矩阵计算这一领域如此非平凡而且有趣. 本节我们建立一个浮点运算的模型并用其讨论浮点的点积、saxpy、矩阵与向量相乘、矩阵与矩阵相乘的误差界. 如需要比我们在此给的还要深入的处理, 见 Higham(1996) 或 Wilkinson(1965). Forsythe and Moler(1967) 以及 Stewart(1973) 的讨论也是很精彩的.

### 2.4.1 浮点数

在计算机进行计算时, 每一算术运算通常都被舍入误差影响. 此误差的产生是因为机器硬件仅能表示实数的一个子集. 我们用 $\mathbf{F}$ 表示该子集且称它的元素为*浮点数*. 按照 Forsythe, Malcolm, and Moler(1997, 第 10~29 页) 的约定, 一个特定的计算机上的浮点数系统可由四个整数来刻画: *基数*$\beta$, *精度*$t$, *指数区间*$[L, U]$. 具体地说, $\mathbf{F}$ 由所有形如

$$f = \pm .d_1d_2\cdots d_t \times \beta^e, 0 \leqslant d_i < \beta, d_1 \neq 0, \quad L \leqslant e \leqslant U$$

的数 $f$ 和零组成. 注意对非零 $f \in \mathbf{F}$, 我们有 $m \leqslant |t| \leqslant M$, 其中

$$m = \beta^{L-1} \text{ 和 } M = \beta^U(1 - \beta^{-t}). \tag{2.4.1}$$

例如，如果 $\beta=2$, $t=3$, $L=0$ 和 $U=2$, 则 $\mathbf{F}$ 的非负元素在图 2.4.1 中用数轴上的竖线所表示. 注意这些浮点数并非等矩分布. $(\beta,t,L,U)$ 的一个典型取值为 $(2,56,-64,64)$.

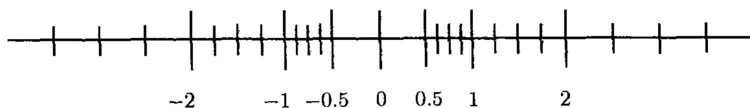



图 2.4.1　浮点数系统例子

### 2.4.2　浮点运算模型

为了给出一给定算法中舍入误差之影响的一般性结论, 有必要给出 $\mathbf{F}$ 中的计算机运算模型. 为此, 定义集合 $G$,

$$G=\{x\in\mathbb{R}:m\leqslant|x|\leqslant M\}\cup\{0\},\tag{2.4.2}$$

以及 $G\to\mathbf{F}$ 的算子 $fl$,

$$fl(x)=\left\{\begin{array}{l}\text{最靠近}x\text{的}c\in\mathbf{F},\text{当}c\text{不唯一时}\\ \text{用截断方式选取}\end{array}\right\}$$

可证明 $fl$ 算子满足

$$fl(x)=x(1+\varepsilon),\quad |\varepsilon|\leqslant\mathbf{u},\tag{2.4.3}$$

其中 $\mathbf{u}$ 是单位舍入, 其定义为

$$\mathbf{u}=\frac{1}{2}\beta^{1-t}.\tag{2.4.4}$$

设 $a$ 和 $b$ 是任意两个浮点数且用“op”表示四种运算 $+,-,\times,\div$ 中的任一种. 如果 $a\ \mathrm{op}\ b\in G$, 则在我们的浮点运算模型中, 我们约定 $(a\ \mathrm{op}\ b)$ 的计算值是 $fl(a\ \mathrm{op}\ b)$. 我们有 $fl(a\ \mathrm{op}\ b)=(a\ \mathrm{op}\ b)(1+\varepsilon)$, 其中 $|\varepsilon|\leqslant\mathbf{u}$. 于是

$$\frac{|fl(a\ \mathrm{op}\ b)-(a\ \mathrm{op}\ b)|}{|a\ \mathrm{op}\ b|}\leqslant\mathbf{u},\quad a\ \mathrm{op}\ b\neq 0.\tag{2.4.5}$$

这表明每个算术运算都有小的相对误差①. 但是, 当涉及一系列运算时, 情况并非如此.

**例 2.4.1**　*设浮点运算中 $\beta=10$, $t=3$, 则 $fl[fl(10^{-4}+1)-1]=0$ 表明相对误差为 1. 另一方面, 精确答案由 $fl[fl(10^{-4}+fl(1-1))]=10^{-4}$ 给出. 故知, 浮点运算并不总是可结合的.*

如果 $a\ \mathrm{op}\ b\notin G$, 则出现了*运算违例*. 当 $|a\ \mathrm{op}\ b|>M$ 或 $|a\ \mathrm{op}\ b|<m$ 时分别称为上溢和下溢. 处理此类情形以及其他违例情形是与机器硬件和系统有关的.

① 有些重要的机器的加法浮点运算满足 $fl(a\pm b)=(1+\varepsilon_1)a\pm(1+\varepsilon_2)b$, 其中 $|\varepsilon_1|,|\varepsilon_2|\leqslant u$. 在这种机器环境下, 不等式 $|fl(a\pm b)-(a\pm b)|\leqslant u|a\pm b|$ 不一定成立.

### 2.4.3 相消

有限精度运算的另一重要方面是*灾难性相消现象*. 精略地说, 这一术语是指用大数相加得到小数时所导致有效位数的大量失去. Forsythe, Malcolm and Moler (1977, 第 14~16 页) 给出的一个著名例子是用 Taylor 级数计算 $e^{-a}$, 其中 $a > 0$. 此方法的舍入误差大约是最大部分和的 $u$ 倍. 对于大数 $a$, 该误差实际上比要求的指数 $e^{-a}$ 还大, 因而无论级数和中用多少项, 计算结果中都没有正确的有效数字. 另一方面, 如果在 $e^a$ 的 Taylor 级数中有足够多项相加, 然后求其倒数, 则可得到满足精度的 $e^{-a}$ 之近似值.

### 2.4.4 绝对值记号

在讨论一些基本矩阵计算的舍入误差分析前, 我们先熟悉一些有用的记号. 设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 我们希望把它的浮点数表示所导致的误差量化. 记 $\boldsymbol{A}$ 的储存值为 $fl(\boldsymbol{A})$, 可知对所有 $i$ 和 $j$ 都有

$$[fl(\boldsymbol{A})]_{ij} = fl(a_{ij}) = a_{ij}(1+\varepsilon_{ij}), \quad |\varepsilon_{ij}| \leqslant \mathbf{u}. \tag{2.4.6}$$

如果采用两个约定则可得到一更好的方式来描述此结果. 设 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都属于 $\mathbb{R}^{m\times n}$, 则

$$\begin{aligned} \boldsymbol{B} = |\boldsymbol{A}| &\Rightarrow b_{ij} = |a_{ij}|, \quad i = 1:m, \quad j = 1:n; \\ \boldsymbol{B} \leqslant \boldsymbol{A} &\Rightarrow b_{ij} < a_{ij}, \quad i = 1:m, \quad j = 1:n. \end{aligned}$$

利用这些记号, 则 (2.4.6) 可写成

$$|fl(\boldsymbol{A}) - \boldsymbol{A}| \leqslant \mathbf{u}|\boldsymbol{A}|.$$

这样一个关系式很容易变成范数不等式, 例如 $\|fl(\boldsymbol{A}) - \boldsymbol{A}\|_1 \leqslant u\|\boldsymbol{A}\|_1$. 但是, 在量化矩阵运算中的舍入误差时, 绝对值记号比范数更有信息性, 这是因为绝对值记号对每一个 $(i,j)$ 元素都有估计.

### 2.4.5 点积的舍入误差

我们以考虑标准的点积所引起的舍入误差来开始有限精度矩阵计算的研究:

$s = 0$
**for** $k = 1:n$
　　$s = s + x_k y_k$ (2.4.7)
**end**

这里 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是 $n\times 1$ 浮点向量.

要试图量化此算法的舍入误差, 我们立刻面临一个记号问题: 计算值与精确值的区分. 当所考虑的计算很明显时, 我们用算子 $fl(\cdot)$ 表示计算值. 这样 $fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y})$ 就表示 (2.4.7) 所输出的计算结果. 我们来估计 $|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}|$ 的界. 如果

$$s_p = fl\left(\sum_{k=1}^{p} x_k y_k\right),$$

则 $s_1 = x_1y_1(1+\delta_1)$, 其中 $|\delta_1| \leqslant u$, 而且对于 $p = 2:n$ 有

$$\begin{aligned} s_p &= fl(s_{p-1} + fl(x_p y_p)) \\ &= (s_{p-1} + x_p y_p(1+\delta_p))(1+\varepsilon_p), \quad |\delta_p|, |\varepsilon_p| \leqslant \mathbf{u}. \end{aligned} \tag{2.4.8}$$

通过简单地代数运算可得

$$fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) = s_n = \sum_{k=1}^{n} x_k y_k (1+\gamma_k),$$

其中

$$(1+\gamma_k) = (1+\delta_k)\prod_{j=k}^{n}(1+\varepsilon_j),$$

这里利用了约定 $\varepsilon_1 = 0$. 于是

$$|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \leqslant \sum_{k=1}^{n} |x_k y_k||\gamma_k|. \tag{2.4.9}$$

为进一步分析, 我们必须用 $u$ 来给出 $|\gamma_k|$ 的界. 下面的结果对此是有用的.

**引理 2.4.1** 如果 $(1+\alpha) = \prod_{k=1}^{n}(1+\alpha_k)$, 其中 $|\alpha_k| \leqslant \mathbf{u}$ 且 $n\mathbf{u} \leqslant 0.1$, 则 $|\alpha| \leqslant 1.01n\mathbf{u}$.

**证明** 见 Higham(1996, p.75). □

把这一结果用于 (2.4.9) 且做“合理”假定 $n \cdot u \leqslant 0.1$, 则得到

$$|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \leqslant 1.01n\mathbf{u}|\boldsymbol{x}|^{\mathrm{T}}|\boldsymbol{y}|. \tag{2.4.10}$$

注意, 在 $|\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \ll |\boldsymbol{x}|^{\mathrm{T}}|\boldsymbol{y}|$ 时 $fl(x^{\mathrm{T}}y)$ 的相对误差可能不会小.

### 2.4.6 量化舍入误差的其他方式

估计引理 2.4.1 中 $\alpha$ 之界的一个较简单但不太精确的方式是说 $|\alpha| \leqslant n\mathbf{u} + O(\mathbf{u}^2)$. 利用此约定, 我们有

$$|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \leqslant n\mathbf{u}|\boldsymbol{x}|^{\mathrm{T}}|\boldsymbol{y}| + O(\mathbf{u}^2). \tag{2.4.11}$$

表示同样结果的其他方式包括

$$|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \leqslant \phi(n)\mathbf{u}|\boldsymbol{x}|^{\mathrm{T}}|\boldsymbol{y}|, \tag{2.4.12}$$

$$|fl(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}) - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}| \leqslant cn\mathbf{u}|\boldsymbol{x}|^{\mathrm{T}}|\boldsymbol{y}|, \tag{2.4.13}$$

其中 (2.4.12) 中的 $\phi(n)$ 是 $n$ 的一个“温和”函数, (2.4.13) 中的 $c$ 是一个量级为 1 的常数.

我们对 (2.4.10)~(2.4.13) 所给出的误差界形式不表示任何的偏好. 这样我们就不必将文献中的误差分析结果改写成某固定形式. 而且, 过份关心误差界的细节与舍入误差分析之"宗旨"是不符的. 正如 Wilkinson(1971, p.567) 所说:

仍然有一种倾向, 把由先验误差分析所得到的精确误差界看得过重. 依我之见, 界本身通常是最不重要的. 此类分析的主要目的在于揭露算法中可能存在的潜在不稳定性, 从而希望由所得到的内在性质来改进算法. 界本身常常是比它能达到的界要弱, 这是因为需要把大量的细节限制到合理的水平, 还因为把误差用矩阵范数来表示所带来的局限性. 一般来说, 先验误差界是不应在实际中用的量. 实用的误差界常常由某后验误差分析决定, 因为这样可充分利用舍入误差的统计分布和矩阵的特殊性质, 如稀疏性等.

重要的是我们要牢记这些观点.

### 2.4.7 点积累加

有些计算机能够用双精度来累加点积. 这就是说, 如果 $x$ 和 $y$ 是长度为 $t$ 尾数的浮点向量, 则 (2.4.7) 中的和 $s$ 是在具有 $2t$ 位尾数的记数器中迭加. 由于两个 $t$ 位的浮点数之积可精确地用双精度变量表示, 只有当 $s$ 写回到单精度内存时才产生舍入误差. 在此情形下, 常常可以断言计算的点积有好的相对误差, 即 $fl(x^{\mathrm{T}}y) = x^{\mathrm{T}}y(1+\delta), |\delta| \approx u$. 因而, 累加点积是很有吸引力的.

### 2.4.8 其他矩阵计算中的舍入误差

很容易证明, 如果 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 是浮点矩阵, $\alpha$ 是浮点数, 则

$$fl(\alpha \boldsymbol{A}) = \alpha \boldsymbol{A} + \boldsymbol{E}, \quad |\boldsymbol{E}| \leqslant \mathbf{u}|\alpha \boldsymbol{A}|, \tag{2.4.14}$$

$$fl(\boldsymbol{A}+\boldsymbol{B}) = (\boldsymbol{A}+\boldsymbol{B}) + \boldsymbol{E}, \quad |\boldsymbol{E}| \leqslant \mathbf{u}|\boldsymbol{A}+\boldsymbol{B}|. \tag{2.4.15}$$

从这两个结果很容易验证计算出来的 saxpy 和外积修正满足

$$fl(\alpha x + y) = \alpha x + y + z, \quad |z| \leqslant \mathbf{u}(2|\alpha x| + |y|) + O(\mathbf{u}^2), \tag{2.4.16}$$

$$fl(\boldsymbol{C} + uv^{\mathrm{T}}) = \boldsymbol{C} + uv^{\mathrm{T}} + \boldsymbol{E}, \quad |\boldsymbol{E}| \leqslant \mathbf{u}(|\boldsymbol{C}| + 2|uv^{\mathrm{T}}|) + O(\mathbf{u}^2). \tag{2.4.17}$$

利用 (2.4.10), 不难证明两个浮点矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的基于点积的乘法满足

$$fl(\boldsymbol{AB}) = \boldsymbol{AB} + \boldsymbol{E}, \quad |\boldsymbol{E}| \leqslant n\mathbf{u}|\boldsymbol{A}||\boldsymbol{B}| + O(\mathbf{u}^2). \tag{2.4.18}$$

基于 gaxpy 以及基于外积的乘法也有同样的结果. 注意, 矩阵乘法并不一定有小的相对误差, 这是因为 $|\boldsymbol{AB}|$ 可能远小于 $|\boldsymbol{A}||\boldsymbol{B}|$, 例如

$$\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -0.99 & 0 \end{bmatrix} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0 \end{bmatrix}.$$

利用到目前所讨论的舍入误差结果, 不难得到一些范数界. 如果我们考查浮点矩阵乘法的 1 范数误差, 则容易从 (2.4.18) 证明

$$\|fl(\boldsymbol{AB}) - \boldsymbol{AB}\|_1 \leqslant n\mathbf{u}\|\boldsymbol{A}\|_1\|\boldsymbol{B}\|_1 + O(\mathbf{u}^2). \tag{2.4.19}$$

### 2.4.9 向前误差分析和向后误差分析

上面给出的舍入误差界都是由*向前误差分析*所导出的. 另一个刻画算法舍入误差的方式可通过称之为*向后误差分析*的技巧来实现. 在此情形下, 舍入误差是关于问题的数据而不是关于解. 作为一个例子, 考虑 $n = 2$ 时的三角形矩阵相乘. 可以证明:

$$fl(\boldsymbol{AB}) = \begin{bmatrix} a_{11}b_{11}(1+\varepsilon_1) & (a_{11}b_{12}(1+\varepsilon_2) + a_{12}b_{22}(1+\varepsilon_3))(1+\varepsilon_4) \\ 0 & a_{22}b_{22}(1+\varepsilon_5) \end{bmatrix},$$

其中 $|\varepsilon_i| \leqslant u(i = 1:5)$. 但是, 如果我们定义

$$\hat{\boldsymbol{A}} = \begin{bmatrix} a_{11} & a_{12}(1+\varepsilon_3)(1+\varepsilon_4) \\ 0 & a_{22}(1+\varepsilon_5) \end{bmatrix},$$

$$\hat{\boldsymbol{B}} = \begin{bmatrix} b_{11}(1+\varepsilon_1) & b_{12}(1+\varepsilon_2)(1+\varepsilon_4) \\ 0 & b_{22} \end{bmatrix},$$

则容易证明 $fl(\boldsymbol{AB}) = \hat{\boldsymbol{A}}\,\hat{\boldsymbol{B}}$. 而且

$$\begin{aligned} \hat{\boldsymbol{A}} &= \boldsymbol{A} + \boldsymbol{E}, \quad |\boldsymbol{E}| \leqslant 2\mathbf{u}|\boldsymbol{A}| + O(\mathbf{u}^2); \\ \hat{\boldsymbol{B}} &= \boldsymbol{B} + \boldsymbol{F}, \quad |\boldsymbol{F}| \leqslant 2\mathbf{u}|\boldsymbol{B}| + O(\mathbf{u}^2). \end{aligned}$$

换句话说, 计算的乘积是稍加扰动后的 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的精确乘积.

### 2.4.10 Strassen 乘积的误差

在 1.3.8 节我们简述了 Strassen(1969) 提出的一种非常规的矩阵相乘方法. 比较该方法和 1.1 节的常规矩阵相乘方法之舍入误差的影响是有益的.

可以证明, Strassen 方法 (算法 1.3.1) 产生的 $\hat{\boldsymbol{C}} = fl(\boldsymbol{AB})$ 满足一个形如 (2.4.19) 的不等式. 这在许多应用中就足够了. 但是, Strassen 方法所产生的 $\hat{\boldsymbol{C}}$ 并不总满足形如 (2.4.18) 的不等式. 为看清这一点, 假定

$$\boldsymbol{A} = \boldsymbol{B} = \begin{bmatrix} 0.99 & 0.0010 \\ 0.0010 & 0.99 \end{bmatrix}$$

以及用 2 位浮点运算执行算法 1.3.1. 其中, 算法计算下列值:

$$\begin{aligned} \hat{P}_3 &= fl(0.99(0.001 - 0.99)) = -0.98, \\ \hat{P}_5 &= fl((0.99 + 0.001)0.99) = 0.98, \\ \hat{c}_{12} &= fl(\hat{P}_3 + \hat{P}_5) = 0.0, \end{aligned}$$

而精确计算有 $c_{12} = 2(0.001)(0.99) = 0.0198$, 所以, 算法 1.3.1 求出的 $\hat{c}_{12}$ 没有一位正确的有效数字. Strassen 方式在此例中有麻烦是因为非对角线元素小而对角线元素大. 注意, 在常规的矩阵乘法中, $b_{12}$ 和 $b_{21}$, $a_{11}$ 和 $a_{12}$ 不会相加, 因而小的非对角线元素在计算中不会被忽视. 事实上, 对上述 $\boldsymbol{A}$ 和 $\boldsymbol{B}$, 常规矩阵乘法将给出 $\hat{c}_{12} = 0.020$.

不能按元素准确地算出 $\hat{C}$, 这在一些情形下是非常严重的缺陷. 例如, 在 Markov 过程中, $a_{ij}, b_{ij}$ 和 $c_{ij}$ 是转移概率, 故是非负的. 如果 $c_{ij}$ 代表所考虑问题中某特别重要的概率, 则精确地计算它就可能是至关重要. 注意, 如果 $\boldsymbol{A} \geqslant 0$ 和 $\boldsymbol{B} \geqslant 0$, 则常规的矩阵乘法计算出的 $\hat{C}$ 有小的相对误差:

$$|\hat{C} - \boldsymbol{C}| \leqslant n\mathbf{u}|\boldsymbol{A}||\boldsymbol{B}| + O(\mathbf{u}^2) = n\mathbf{u}|\boldsymbol{C}| + O(\mathbf{u}^2).$$

上式由 (2.4.18) 得到. 此关系式对 Strassen 方法并不一定对, 因而对于非负矩阵相乘需要比较精确计算 $\hat{c}_{ij}$ 时, 算法 1.3.1 并不是很好的.

从上面讨论延伸, 我们得到两个很显然却重要的结论:

- 不同的方法计算同一量可产生很不同的结果;
- 一个算法能否产生满意的结果取决于所述问题的类型以及用户的目标.

这些观点将在后面的各章中予以阐明, 它们与算法稳定性和问题条件等概念是紧密相关的.

## 习 题

**2.4.1** 证明, 如果 (2.4.7) 应用于 $y = x$, 则 $fl(x^{\mathrm{T}}x) = x^{\mathrm{T}}x(1+\alpha)$, 其中 $|\alpha| \leqslant n\mathbf{u} + O(\mathbf{u}^2)$.

**2.4.2** 证明 (2.4.3).

**2.4.3** 证明, 如果 $\boldsymbol{E} \in \mathbb{R}^{m\times n}(m > n)$, 则 $\||\boldsymbol{E}|\|_2 \leqslant \sqrt{n}\|\boldsymbol{E}\|_2$. 此结果对于从绝对值界导出范数界是很有用的.

**2.4.4** 假设存在一平方根函数满足 $fl(\sqrt{x}) = \sqrt{x}(1+\varepsilon)$, 其中 $|\varepsilon| \leqslant \mathbf{u}$. 给出一个计算 $\|x\|_2$ 的算法并估计误差界.

**2.4.5** 假设 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是 $n \times n$ 上三角浮点阵, 如果 $\hat{\boldsymbol{C}} = fl(\boldsymbol{AB})$ 是用 1.1 节中的常规算法所计算的, 问是否有 $\hat{\boldsymbol{C}} = \hat{\boldsymbol{A}}\hat{\boldsymbol{B}}$, 其中 $\hat{\boldsymbol{A}}$ 和 $\hat{\boldsymbol{B}}$ 分别靠近 $\boldsymbol{A}$ 和 $\boldsymbol{B}$?

**2.4.6** 假定 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 是 $n \times n$ 浮点矩阵, $\boldsymbol{A}$ 非奇且满足 $\||\boldsymbol{A}^{-1}||\boldsymbol{A}|\|_\infty = \tau$. 证明: 如果 $\hat{\boldsymbol{C}} = fl(\boldsymbol{AB})$ 是用 1.1 节中的任何算法求得的, 则存在 $\hat{\boldsymbol{B}}$ 使得 $\hat{\boldsymbol{C}} = \boldsymbol{A}_1\hat{\boldsymbol{B}}$ 且 $\|\hat{\boldsymbol{B}} - \boldsymbol{B}\|_\infty \leqslant n\mathbf{u}\tau\|\boldsymbol{B}\|_\infty + O(\mathbf{u}^2)$.

**2.4.7** 证明 (2.4.18).

## 本节注释与参考文献

对于舍入误差的一般介绍, 我们推荐:

J. H. Wilkinson(1963). *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, NJ.

J. H. Wilkinson(1971). "Modern Error Analysis," *SIAM Review* 13, 548–568.

D. Kahaner, C. B. Moler, and S. Nash(1988). *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ.

F. Chaitin-Chatelin and V. Frayseé(1996). *Lectures on Finite Precision Computations*, SIAM Publications, Philadelphia.

关于涉及区间分析的误差分析, 舍入误差的统计模型建立, 以及误差本身的运算等的最新进展, 可见:

T. E. Hull and J. R. Swensen(1966). "Tests of Probabilistic Models for Propagation of Roundoff Errors," *Comm. ACM.* 9. 108–113.

J. Larson and A. Sameh(1978). "Efficient Calculation of the Effects of Roundoff Errors," *ACM Trans. Math. Soft.* 4, 228–236.

W. Miller and D. Spooner(1978). "Software for Roundoff Analysis, II," *ACM Trans. Math. Soft.* 4, 369–390.

J. M. Yohe(1979). "Software for Interval Arithmetic: A Reasonable Portable Package," *ACM Trans. Math. Soft.* 5, 50–63.

任何真正做软件开发的人必须对浮点运算有透彻的了解. 了解这方面知识的一个好的方法是学习 IEEE 的浮点标准.

D. Goldberg(1991). "What Every Computer Scientist Should Know About Floating Point Arithmetic," *ACM Surveys* 23, 5–48.

也可见:

R. P. Brent(1978). "A Fortran Multiple Precision Arithmetic Package," *ACM Trans. Math. Soft.* 4, 57–70

R. P. Brent(1978). "Algorithm 524 MP, a Fortran Multiple Precision Arithmetic Package," *ACM Trans. Math. Soft.* 4, 71–81.

J. W. Demmel(1984). "Underflow and the Reliability of Numerical Software," *SIAM J. Sci. and Stat. Comp.* 5, 887–919.

U. W. Kulisch and W. L. Miranker(1986). "The Arithmetic of the Digital Computer," *SIAM Review* 28, 1–40.

W. J. Cody(1988). "ALGORITHM 665 MACHAR: A Subroutine to Dynamically Determine Machine Parameters," *ACM Trans. Math. Soft.* 14, 303–311.

D. H. Bailey, H. D. Simon, J. T. Barton, M. J. Fouts(1989). "Floating Point Arithmetic in Future Supercomputers." *Int'l J. Supercomputing Appl.* 3, 86–90.

D. H. Bailey(1993). "Algorithm 719: Multiprecision Translation and Execution of FORTRAN Programs," *ACM Trans. Math. Soft.* 19, 288–319.

开发高性能软件的技巧, 甚至对于"简单"问题, 也是非常重要. 一个很好的例子是计算 2 范数的子程序设计.

J. M. Blue(1978), "A Portable FORTRAN Program to Find the Euclidean Norm of a

Vector," *ACM Trans. Math. Soft.* 4, 15–23.

关于 Strassen 算法以及其他线性代数"快速"方法的分析, 可见:

R. P. Brent(1970). "Error Analysis of Algorithms for Matrix Multiplication and Triangular Decomposition Using Winograd's Identity," *Numer. Math.* 16, 145–156.

W. Miller(1975). "Computational Complexity and Numerical Stability," *SIAM J. Computing* 4, 97–107.

N. J. Higham(1992). "Stability of a Method for Multiplying Complex Matrices with Three Real Matrix Multiplications," *SIAM J. Matrix Anal. Appl.* 13, 681–687.

J. W. Demmel and N.J. Higham(1992). "Stability of Block Algorithms with Fast Level-3 BLAS," *ACM Trans. Math. Soft.* 18, 274–291.

## 2.5 正交化与 SVD

正交化在矩阵计算中起非常重要的作用. 在给出一些定义之后我们证明极其有用的奇异值分解 (SVD). 除了其他作用, SVD 使我们能巧妙地处理矩阵秩的问题. 秩的概念, 在精确运算下虽然十分清楚, 但在有舍入误差以及模糊数据情况下就很复杂了. 利用 SVD, 我们可以引入实用的数值秩的概念.

### 2.5.1 正交性

如果对任意 $i \neq j$ 都有 $\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j = 0$, 则一组向量 $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_p\} \in \mathbb{R}^m$ *是正交的*. 如果 $\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j = \delta_{ij}$, 则称为规范正交. 直观地说, 正交向量是最无关的, 因为它们指向完全不同的方向.

$\mathbb{R}^m$ 中一组子空间 $S_1, \cdots, S_p$ 称为*相互正交的*, 如果对 $i \neq j$ 都有 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} = 0(\boldsymbol{x} \in S_i, \boldsymbol{y} \in S_j)$. 子空间 $S \subseteq \mathbb{R}^m$ 的*正交补*由

$$S^{\perp} = \{\boldsymbol{y} \in \mathbb{R}^m : \boldsymbol{y}^{\mathrm{T}}\boldsymbol{x} = 0, \ \forall \boldsymbol{x} \in S\}$$

所定义, 不难证明 $\mathrm{ran}(\boldsymbol{A})^{\perp} = \mathrm{null}(\boldsymbol{A}^{\mathrm{T}})$. 如果向量 $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k$ 规范正交且张成 $\mathbb{R}^m$ 中的子空间 $S$, 则它们形成 $S$ 的一组规范正交基.

如果 $\boldsymbol{Q} \in \mathbb{R}^{m \times m}$ 满足 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} = \boldsymbol{I}$, 则称其为正交的. 如果 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_m]$ 是正交的, 则 $\boldsymbol{q}_i$ 形成 $\mathbb{R}^m$ 的规范正交基. 总可以把一个基扩充到 $\mathbb{R}^m$ 上的一组完整的规范正交基.

**定理 2.5.1** *如果 $\boldsymbol{V}_1 \in \mathbb{R}^{n \times r}$ 具有规范正交列, 则存在 $\boldsymbol{V}_2 \in \mathbb{R}^{n \times (n-r)}$ 使得*

$$\boldsymbol{V} = [\boldsymbol{V}_1 \quad \boldsymbol{V}_2]$$

*是正交的. 注意到 $\mathrm{ran}(\boldsymbol{V}_1)^{\perp} = \mathrm{ran}(\boldsymbol{V}_2)$.*

**证明** 这是初等线性代数的基本结果, 它也是我们在 5.2 节中给出的 QR 分解的一个推论. □

### 2.5.2 范数和正交变换

2 范数在正交变换下是不变的, 这是因为若 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}=\boldsymbol{I}$, 则 $\|\boldsymbol{Q}\boldsymbol{x}\|_2^2=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x}=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}=\|\boldsymbol{x}\|_2^2$. 矩阵的 2 范数和 Frobenius 范数关于正交变换也是不变的. 特别地, 对于维数适当的正交矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$, 不难证明

$$\|\boldsymbol{QAZ}\|_F=\|\boldsymbol{A}\|_F, \tag{2.5.1}$$

$$\|\boldsymbol{QAZ}\|_2=\|\boldsymbol{A}\|_2. \tag{2.5.2}$$

### 2.5.3 奇异值分解

上两节介绍的有关范数的理论可用来证明极其有用的奇异值分解.

**定理 2.5.2(奇异值分解 (SVD))** 设 $\boldsymbol{A}$ 是实 $m\times n$ 矩阵, 则必存在正交矩阵

$$\boldsymbol{U}=[\boldsymbol{u}_1,\cdots,\boldsymbol{u}_m]\in\mathbb{R}^{m\times m}\quad 和 \quad \boldsymbol{V}=[\boldsymbol{v}_1,\cdots,\boldsymbol{v}_n]\in\mathbb{R}^{n\times n}$$

使得

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{AV}=\mathrm{diag}(\sigma,\cdots,\sigma_p)\in\mathbb{R}^{m\times n},\quad p=\min\{m,n\},$$

其中 $\sigma_1\geqslant\sigma_2\geqslant\cdots\geqslant\sigma_p\geqslant 0$.

**证明** 令 $\boldsymbol{x}\in\mathbb{R}^n$ 和 $\boldsymbol{y}\in\mathbb{R}^m$ 是满足 $\boldsymbol{Ax}=\sigma\boldsymbol{y}(\sigma=\|\boldsymbol{A}\|_2)$ 的 2 范数单位向量. 从定理 2.5.1 知, 存在 $\boldsymbol{V}_2\in\mathbb{R}^{n\times(n-1)}$ 和 $\boldsymbol{U}_2\in\mathbb{R}^{m\times(m-1)}$, 使得 $\boldsymbol{V}=[\boldsymbol{x}\quad \boldsymbol{V}_2]\in\mathbb{R}^{n\times n}$ 和 $\boldsymbol{U}=[\boldsymbol{y}\quad \boldsymbol{U}_2]\in\mathbb{R}^{m\times m}$ 是正交的. 不难证明 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{AV}$ 有如下结构:

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{AV}=\begin{bmatrix}\sigma & \boldsymbol{w}^{\mathrm{T}}\\ \boldsymbol{0} & \boldsymbol{B}\end{bmatrix}\equiv\boldsymbol{A}_1.$$

由于

$$\left\|\boldsymbol{A}_1\left(\begin{bmatrix}\sigma\\ \boldsymbol{w}\end{bmatrix}\right)\right\|_2^2\geqslant(\sigma^2+\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w})^2,$$

我们有 $\|\boldsymbol{A}_1\|_2^2\geqslant(\sigma^2+\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w})$. 但 $\sigma^2=\|\boldsymbol{A}\|_2^2=\|\boldsymbol{A}_1\|_2^2$, 故必有 $w=0$. 用很显然的归纳法即可证明定理. □

$\sigma_i$ 是 $\boldsymbol{A}$ 的奇异值, 向量 $\boldsymbol{u}_i$ 和 $\boldsymbol{v}_i$ 分别是第 $i$ 个左奇异向量和第 $i$ 个右奇异向量, 通过比较 $\boldsymbol{AV}=\boldsymbol{U\Sigma}$ 以及 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U}=\boldsymbol{V\Sigma}^{\mathrm{T}}$ 的对应列容易验证

$$\left.\begin{aligned}\boldsymbol{Av}_i&=\sigma_i\boldsymbol{u}_i\\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_i&=\sigma_i\boldsymbol{v}_i\end{aligned}\right\}i=1:\min\{m,n\}.$$

采用如下表示奇异值的记号是方便的:

$$\begin{aligned}\sigma_i(\boldsymbol{A})&=\boldsymbol{A}的第i大奇异值,\\ \sigma_{\max}(\boldsymbol{A})&=\boldsymbol{A}的最大奇异值,\\ \sigma_{\min}(\boldsymbol{A})&=\boldsymbol{A}的最小奇异值.\end{aligned}$$

矩阵 $\boldsymbol{A}$ 的奇异值正好是由 $E=\{\boldsymbol{Ax}:\|\boldsymbol{x}\|_2=1\}$ 所定义的超椭圆体 $E$ 的半轴之长.

**例 2.5.1**

$$\boldsymbol{A}=\begin{bmatrix}0.96 & 1.72\\ 2.28 & 0.96\end{bmatrix}=\boldsymbol{U}\Sigma\boldsymbol{V}^{\mathrm{T}}$$
$$=\begin{bmatrix}0.6 & -0.8\\ 0.8 & 0.6\end{bmatrix}\begin{bmatrix}3 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}0.8 & 0.6\\ 0.6 & -0.8\end{bmatrix}^{\mathrm{T}}.$$

SVD 深刻揭露了矩阵的结构. 设 $\boldsymbol{A}$ 的 SVD 由定理 2.5.2 给出, 我们由

$$\sigma_1 \geqslant \cdots \geqslant \sigma_r > \sigma_{r+1} = \cdots = \sigma_p = 0$$

定义 $r$, 则

$$\operatorname{rank}(\boldsymbol{A}) = r, \tag{2.5.3}$$
$$\operatorname{null}(\boldsymbol{A}) = \operatorname{span}\{\boldsymbol{v}_{r+1},\cdots,\boldsymbol{v}_n\}, \tag{2.5.4}$$
$$\operatorname{ran}(\boldsymbol{A}) = \operatorname{span}\{\boldsymbol{u}_1,\cdots,\boldsymbol{u}_r\}, \tag{2.5.5}$$

并且有 SVD*展开*

$$\boldsymbol{A}=\sum_{i=1}^{r}\sigma_i\boldsymbol{u}_i\boldsymbol{v}_i^{\mathrm{T}}. \tag{2.5.6}$$

许多 2 范数和 Frobenius 范数性质与 SVD 有关. 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 则

$$\|\boldsymbol{A}\|_F^2=\sigma_1^2+\cdots+\sigma_p^2,\quad p=\min\{m,n\}, \tag{2.5.7}$$
$$\|\boldsymbol{A}\|_2=\sigma_1, \tag{2.5.8}$$
$$\min_{\boldsymbol{x}\neq 0}\frac{\|\boldsymbol{Ax}\|_2}{\|\boldsymbol{x}\|_2}=\sigma_n\quad (m\geqslant n). \tag{2.5.9}$$

### 2.5.4 细 SVD

设 $\boldsymbol{A}=\boldsymbol{U\Sigma V}^{\mathrm{T}}\in\mathbb{R}^{m\times n}$ 是 $\boldsymbol{A}$ 的 SVD 且 $m\geqslant n$. 则

$$\boldsymbol{A}=\boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}^{\mathrm{T}},$$

其中

$$U_1=U(:,1:n)=[u_1,\cdots,u_n]\in\mathbb{R}^{m\times n},$$
$$\boldsymbol{\Sigma}_1=\Sigma(1:n,1:n)=\operatorname{diag}(\sigma_1,\cdots,\sigma_n)\in\mathbb{R}^{n\times n}.$$

我们把这个常用的 SVD 的缩小形式称为细 SVD.

### 2.5.5 秩亏损与 SVD

SVD 最有价值的方面之一是它帮我们很好地处理矩阵秩的概念. 线性代数中的许多定理具有形式:“如果这样那样的一个矩阵满秩, 则这样那样的一个性质成立.” 这类结果虽然简洁和优美, 但出现几乎秩亏损时不能帮助我们解决所碰到的

数值困难. 舍入误差和模糊数据使得秩的确定很困难. 事实上, 对很小的 $\varepsilon$, 我们可能对矩阵的 $\varepsilon$ 秩感兴趣, 其定义为

$$\operatorname{rank}(\boldsymbol{A},\varepsilon)=\min_{\|\boldsymbol{A}-\boldsymbol{B}\|_2\leqslant\varepsilon}\operatorname{rank}(\boldsymbol{B}).$$

这样, 当实验室得到的 $\boldsymbol{A}$ 的每一元素 $a_{ij}$ 具有精度 $\pm0.001$ 时, 考查 $\operatorname{rank}(\boldsymbol{A}, 0.001)$ 就是很合理的.基于同样的考虑, 设 $\boldsymbol{A}$ 是 $m\times n$ 浮点矩阵, 当 $\operatorname{rank}(\boldsymbol{A},\varepsilon)<\min\{m,n\}$ $(\varepsilon=u\|\boldsymbol{A}\|_2)$ 时, 则可认为 $\boldsymbol{A}$ 是*数值秩亏损的*.

数值秩亏损和 $\varepsilon$ 秩由 SVD 很好地刻画, 因为奇异值可表明一个给定矩阵与比其秩低的矩阵之靠近程度.

**定理 2.5.3** 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 的 SVD 由定理 2.5.2 给出. 如果 $k<r=\operatorname{rank}(\boldsymbol{A})$ 且

$$\boldsymbol{A}_k=\sum_{i=1}^{k}\sigma_i\boldsymbol{u}_i\boldsymbol{v}_i^{\mathrm{T}},\tag{2.5.10}$$

则

$$\min_{\operatorname{rank}(\boldsymbol{B})=k}\|\boldsymbol{A}-\boldsymbol{B}\|_2=\|\boldsymbol{A}-\boldsymbol{A}_k\|_2=\sigma_{k+1}.\tag{2.5.11}$$

**证明** 由于 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}_k\boldsymbol{V}=\operatorname{diag}(\sigma_1,\cdots,\sigma_k,0,\cdots,0)$, 从 $\operatorname{rank}(\boldsymbol{A}_k)=k$ 和 $U^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{A}_k)\boldsymbol{V}=\operatorname{diag}(0,\cdots,0,\sigma_{k+1},\cdots,\sigma_p)$ 可知 $\|\boldsymbol{A}-\boldsymbol{A}_k\|_2=\sigma_{k+1}$.

现假定对某一 $\boldsymbol{B}\in\mathbb{R}^{m\times n}$ 有 $\operatorname{rank}(\boldsymbol{B})=k$. 则存在规范正交向量 $\boldsymbol{x}_1,\cdots,\boldsymbol{x}_{n-k}$ 使得 $\operatorname{null}(\boldsymbol{B})=\operatorname{span}\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_{n-k}\}$. 利用维数可知

$$\operatorname{span}\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_{n-k}\}\cap\operatorname{span}\{\boldsymbol{v}_1,\cdots,\boldsymbol{v}_{k+1}\}\neq\{0\}.$$

令 $\boldsymbol{z}$ 是这一交集中的一个单位 2 范数向量. 利用 $\boldsymbol{B}\boldsymbol{z}=0$ 和

$$\boldsymbol{A}\boldsymbol{z}=\sum_{i=1}^{k+1}\sigma_i(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{z})\boldsymbol{u}_i,$$

我们有

$$\|\boldsymbol{A}-\boldsymbol{B}\|_2^2\geqslant\|(\boldsymbol{A}-\boldsymbol{B})\boldsymbol{z}\|_2^2=\|\boldsymbol{A}\boldsymbol{z}\|_2^2=\sum_{i=1}^{k+1}\sigma_i^2(\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{z})^2\geqslant\sigma_{k+1}^2.$$

于是定理得证. □

定理 2.5.3 表明 $\boldsymbol{A}$ 的最小奇异值是从 $\boldsymbol{A}$ 到所有秩亏损矩阵集合之 2 范数距离. 还可知 $\mathbb{R}^{m\times n}$ 中满秩矩阵的集合是开的而且是稠密的.

最后, 如果 $r_\varepsilon=\operatorname{rank}(\boldsymbol{A},\varepsilon)$, 则

$$\sigma_1\geqslant\cdots\geqslant\sigma_{r_\varepsilon}>\varepsilon\geqslant\sigma_{r_\varepsilon+1}\geqslant\cdots\geqslant\sigma_p,\quad p=\min\{m,n\}.$$

在 5.5 节和 12.2 节将进一步讨论数值秩.

### 2.5.6 酉矩阵

在复数域上, 对应于正交矩阵的是酉矩阵. 确定地说, 如果 $\boldsymbol{Q} \in \mathbb{C}^{n\times n}$ 满足 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^{\mathrm{H}} = \boldsymbol{I}$, 则称其为酉矩阵. 酉矩阵保持 2 范数. 复矩阵的 SVD 涉及酉矩阵. 设 $\boldsymbol{A} \in \mathbb{C}^{m\times n}$, 则存在酉矩阵 $\boldsymbol{U} \in \mathbb{C}^{m\times m}$ 和 $\boldsymbol{V} \in \mathbb{C}^{n\times n}$ 使得

$$\boldsymbol{U}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{V} = \mathrm{diag}(\sigma_1, \cdots, \sigma_p) \in \mathbb{R}^{m\times n}, \quad p = \min\{m, n\},$$

其中 $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_p \geqslant 0$.

## 习　题

**2.5.1**　证明: 如果 $\boldsymbol{S}$ 是实的且 $\boldsymbol{S}^{\mathrm{T}} = -\boldsymbol{S}$, 则 $\boldsymbol{I} - \boldsymbol{S}$ 是非奇异且矩阵 $(\boldsymbol{I} - \boldsymbol{S})^{-1}(\boldsymbol{I} + \boldsymbol{S})$ 是正交的. 这称之为 $\boldsymbol{S}$ 的 Cayley 变换.

**2.5.2**　证明一个正交的三角形矩阵必是对角的.

**2.5.3**　证明: 设 $\boldsymbol{Q}_1, \boldsymbol{Q}_2 \in \mathbb{R}^{n\times n}$ 且 $\boldsymbol{Q} = \boldsymbol{Q}_1 + \mathrm{i}\boldsymbol{Q}_2$ 是酉矩阵, 则 $2n \times 2n$ 实矩阵

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Q}_1 & -\boldsymbol{Q}_2 \\ \boldsymbol{Q}_2 & \boldsymbol{Q}_1 \end{bmatrix}$$

是正交的.

**2.5.4**　证明 (2.5.3)~(2.5.9).

**2.5.5**　证明

$$\sigma_{\max}(\boldsymbol{A}) = \max_{\boldsymbol{y}\in\mathbb{R}^m, \boldsymbol{x}\in\mathbb{R}^n} \frac{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2}.$$

**2.5.6**　对于 $2 \times 2$ 矩阵 $\boldsymbol{A} = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$, 导出依赖于 $w, x, y$ 和 $z$ 的表达式 $\sigma_{\max}(\boldsymbol{A})$ 和 $\sigma_{\min}(\boldsymbol{A})$.

**2.5.7**　证明 $\mathbb{R}^{m\times n}$ 中任何矩阵都是满秩矩阵的极限.

**2.5.8**　证明, 如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 的秩为 $n$, 则 $\|\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\|_2 = 1$.

**2.5.9**　什么矩阵是在 Frobenius 范数意义下最靠近

$$\boldsymbol{A} = \begin{bmatrix} 1 & M \\ 0 & 1 \end{bmatrix}$$

的秩 1 矩阵?

**2.5.10**　证明: 如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, 则 $\|\boldsymbol{A}\|_F \leqslant \sqrt{\mathrm{rank}(\boldsymbol{A})}\|\boldsymbol{A}\|_2$, 因而将 (2.3.7) 加强了.

## 本节注释与参考文献

Forsythe and Moler(1967) 很好地描述了 SVD 在 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 问题分析中所起的作用. 他们关于分解的证明比我们的要更传统, 利用了对称矩阵的特征值理论. 关于 SVD 的历史性文献有:

E. Beltrami(1873). "Sulle Funzioni Bilineari," *Gionale di Mathematiche* 11, 98–106.

C. Eckart and G. Young(1939). "A Principal Axis Transformation for Non-Hermitian Matrices," *Bull Amer. Math. Soc.* 45, 118–121.

G. W. Stewart(1993). "On the Early History of the Singular Value Decomposition," *SIAM Review* 35, 551–566.

科学计算中最重要的进展之一是, 在需要对矩阵秩很好处理的实用领域中, SVD 的日益增加的应用. 实用的范围是显著的. 最有趣之一是:

C. B. Moler and D. Morrison(1983). "Singular Value Analysis of Cryptograms," *Amer. Math. Monthly* 90, 78–87.

关于将 SVD 推广至 Hilbert 空间, 可见:

I. C. Gohberg and M. G. Krein(1969). *Introduction to the Theory of Linear Non-Self Adjoint Operators*, Amer. Math. Soc., Providence, R.I.

F. Smithies(1970). *Integral Equations*, Cambridge University Press, Cambridge.

当扰动矩阵有约束时, 类似于定理 2.5.3 将矩阵降秩的讨论可见:

J. W. Demmel(1987). "The smallest perturbation of a submatrix which lowers the rank and constrained total least squares problems," *SIAM J. Numer. Anal.* 24, 199–206.

G. H. Golub, A. Hoffman, and G. W. Stewart(1988). "A Generalization of the Eckart-Young-Mirsky Approximation Theorem," *Lin. Alg. and Its Applic.* 88/89, 317–328.

G. A. Watson(1988). "The Smallest Perturbation of a Submatrix which Lowers the Rank of the Matrix," *IMA J. Numer. Anal.* 8, 295–304.

## 2.6 投影与 CS 分解

如果计算的任务是求一个矩阵或一个向量, 则范数在判断答案的精度或者是度量迭代的进程中是有用的. 如果计算的任务是求一个子空间, 进行类似的判断和度量就需要量化两个子空间的距离. 关于这方面, 正交投影尤为重要. 给出基本概念之后我们将讨论 CS 分解. 这是一类似于 SVD 的分解, 在比较两个子空间时很方便, 我们从正交投影的概念开始.

### 2.6.1 正交投影

设 $S \subseteq \mathbb{R}^{n\times n}$ 是子空间, 如果 $\boldsymbol{P} \in \mathbb{R}^{n\times m}$ 满足 $\mathrm{ran}(\boldsymbol{P}) = \boldsymbol{S}$, $\boldsymbol{P}^2 = \boldsymbol{P}$ 和 $\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{P}$, 则称 $\boldsymbol{P}$ 是向 $\boldsymbol{S}$ 上的**正交投影**. 从定义容易得知, 如果 $\boldsymbol{x} \in \mathbb{R}^n$, 则 $\boldsymbol{Px} \in S$ 且 $(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{x} \in S^{\perp}$.

如果 $\boldsymbol{P}_1$ 和 $\boldsymbol{P}_2$ 都是正交投影, 则对任何 $\boldsymbol{z} \in \mathbb{R}^n$ 有

$$\|(\boldsymbol{P}_1-\boldsymbol{P}_2)\boldsymbol{z}\|_2^2 = (\boldsymbol{P}_1\boldsymbol{z})^{\mathrm{T}}(\boldsymbol{I}-\boldsymbol{P}_2)\boldsymbol{z} + (\boldsymbol{P}_2\boldsymbol{z})^{\mathrm{T}}(\boldsymbol{I}-\boldsymbol{P}_1)\boldsymbol{z}.$$

如果 $\mathrm{ran}(\boldsymbol{P}_1) = \mathrm{ran}(\boldsymbol{P}_2) = S$, 则上式右端项为零, 故知一个子空间的正交投影是唯一的. 如果 $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k]$ 的列是子空间 $S$ 的一组规范正交基, 则容易证明 $\boldsymbol{P} = \boldsymbol{V}\boldsymbol{V}^{\mathrm{T}}$ 是向 $S$ 上的唯一的正交投影. 特别地, 如果非零向量 $\boldsymbol{v} \in \mathbb{R}^m$, 则 $\boldsymbol{P} = \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}/\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}$ 是向 $S = \mathrm{span}\{v\}$ 的正交投影.

### 2.6.2 SVD 相关的投影

有几个重要的正交投影与奇异值分解有关. 设 $\boldsymbol{A}=\boldsymbol{U\Sigma V}^{\mathrm{T}}\in\mathbb{R}^{m\times n}$ 是 $\boldsymbol{A}$ 的 SVD 且 $r=\mathrm{rank}(\boldsymbol{A})$. 如果 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的分划为

$$\boldsymbol{U}=[\underset{r}{\boldsymbol{U}_r}\quad\underset{m-r}{\widetilde{\boldsymbol{U}}_r}],\qquad \boldsymbol{V}=[\underset{r}{\boldsymbol{V}_r}\quad\underset{n-r}{\widetilde{\boldsymbol{V}}_r}],$$

则

$\boldsymbol{V}_r\boldsymbol{V}_r^{\mathrm{T}}$ 是向 $\mathrm{null}(\boldsymbol{A})^{\perp}=\mathrm{ran}(\boldsymbol{A}^{\mathrm{T}})$上的正交投影.
$\widetilde{\boldsymbol{V}}_r\widetilde{\boldsymbol{V}}_r^{\mathrm{T}}$ 是向 $\mathrm{null}(\boldsymbol{A})$ 上的正交投影.
$\boldsymbol{U}_r\boldsymbol{U}_r^{\mathrm{T}}$ 是向 $\mathrm{ran}(\boldsymbol{A})$ 上的正交投影.
$\widetilde{\boldsymbol{U}}_r\widetilde{\boldsymbol{U}}_r^{\mathrm{T}}$ 是向 $\mathrm{ran}(\boldsymbol{A})^{\perp}=\mathrm{null}(\boldsymbol{A})^{\mathrm{T}}$ 上的正交投影.

### 2.6.3 子空间之间的距离

子空间与正交投影的一一对应使我们能导出子空间之间距离的概念. 设 $S_1$ 和 $S_2$ 是 $\mathbb{R}^n$ 中满足 $\dim(S_1)=\mathrm{din}(S_2)$ 的两个子空间. 我们定义这两个子空间之**距离**为

$$\mathrm{dist}(S_1,S_2)=\|\boldsymbol{P}_1-\boldsymbol{P}_2\|_2,\tag{2.6.1}$$

其中 $\boldsymbol{P}_i$ 是向 $S_i$ 上的正交投影. 两个子空间的距离可以用某正交矩阵的分块来刻画.

**定理 2.6.1** 设

$$\boldsymbol{W}=[\underset{k}{\boldsymbol{W}_1}\quad\underset{n-k}{\boldsymbol{W}_2}],\qquad \boldsymbol{Z}=[\underset{k}{\boldsymbol{Z}_1}\quad\underset{n-k}{\boldsymbol{Z}_2}]$$

*都是 $n\times n$ 正交矩阵, 如果 $S_1=\mathrm{ran}(\boldsymbol{W}_1)$ 和 $S_2=\mathrm{ran}(\boldsymbol{Z}_1)$, 则*

$$\mathrm{dist}(S_1,S_2)=\|\boldsymbol{W}_1^{\mathrm{T}}\boldsymbol{Z}_2\|_2=\|\boldsymbol{Z}_1^{\mathrm{T}}\boldsymbol{W}_2\|_2.$$

**证明**

$$\begin{aligned}\mathrm{dist}(S_1,S_2)&=\|\boldsymbol{W}_1\boldsymbol{W}_1^{\mathrm{T}}-\boldsymbol{Z}_1\boldsymbol{Z}_1^{\mathrm{T}}\|_2\\&=\|\boldsymbol{W}^{\mathrm{T}}(\boldsymbol{W}_1\boldsymbol{W}_1^{\mathrm{T}}-\boldsymbol{Z}_1\boldsymbol{Z}_1^{\mathrm{T}})\boldsymbol{Z}\|_2\\&=\left\|\begin{bmatrix}\boldsymbol{0}&\boldsymbol{W}_1^{\mathrm{T}}\boldsymbol{Z}_2\\-\boldsymbol{W}_2^{\mathrm{T}}\boldsymbol{Z}_1&\boldsymbol{0}\end{bmatrix}\right\|_2.\end{aligned}$$

注意到 $\boldsymbol{W}_2^{\mathrm{T}}\boldsymbol{Z}_1$ 和 $\boldsymbol{W}_1^{\mathrm{T}}\boldsymbol{Z}_2$ 是正交矩阵

$$\boldsymbol{Q}=\begin{bmatrix}\boldsymbol{Q}_{11}&\boldsymbol{Q}_{12}\\\boldsymbol{Q}_{21}&\boldsymbol{Q}_{22}\end{bmatrix}\equiv\begin{bmatrix}\boldsymbol{W}_1^{\mathrm{T}}\boldsymbol{Z}_1&\boldsymbol{W}_1^{\mathrm{T}}\boldsymbol{Z}_2\\\boldsymbol{W}_2^{\mathrm{T}}\boldsymbol{Z}_1&\boldsymbol{W}_2^{\mathrm{T}}\boldsymbol{Z}_2\end{bmatrix}=\boldsymbol{W}^{\mathrm{T}}\boldsymbol{Z}$$

的子矩阵. 我们需要证明 $\|\boldsymbol{Q}_{21}\|_2=\|\boldsymbol{Q}_{12}\|_2$. 由于 $\boldsymbol{Q}$ 是正交矩阵, 故知

$$\boldsymbol{Q}\begin{bmatrix}\boldsymbol{x}\\\boldsymbol{0}\end{bmatrix}=\begin{bmatrix}\boldsymbol{Q}_{11}\boldsymbol{x}\\\boldsymbol{Q}_{21}\boldsymbol{x}\end{bmatrix},$$

即

$$1 = \|Q_{11}x\|_2^2 + \|Q_{21}x\|_2^2.$$

对 $\mathbb{R}^k$ 中所有单位 2 范数向量 $x$ 都成立. 于是

$$\begin{aligned}\|Q_{21}\|_2^2 &= \max_{\|x\|_2=1} \|Q_{21}x\|_2^2 = 1 - \min_{\|x\|_2=1} \|Q_{11}x\|_2^2 \\ &= 1 - \sigma_{\min}(Q_{11})^2.\end{aligned}$$

类似地, 讨论 $Q^{\mathrm{T}}$(也是正交矩阵), 可以证明

$$\|Q_{12}^{\mathrm{T}}\|_2^2 = 1 - \sigma_{\min}(Q_{11}^{\mathrm{T}})^2,$$

因而

$$\|Q_{12}\|_2^2 = 1 - \sigma_{\min}(Q_{11})^2.$$

所以 $\|Q_{21}\|_2 = \|Q_{12}\|_2$. □

注意, 当 $S_1$ 和 $S_2$ 是 $\mathbb{R}^n$ 中同维的子空间时, 则有

$$0 \leqslant \mathrm{dist}(S_1, S_2) \leqslant 1.$$

上式当 $S_1 = S_2$ 时左边等号成立, 当 $S_1 \cap S_2^{\perp} \neq \{0\}$ 时右边等号成立.

对上面给出的 $Q$ 矩阵进行分块形式的更精细分析, 有助于更深入了解子空间之距离. 这需要对正交矩阵进行特殊的 SVD 型分解.

### 2.6.4　CS 分解

正交矩阵划分成的 $2 \times 2$ 块与 SVD 密切相关. 这是 CS 分解的基本点. 我们先证明一个很有用的特例.

**定理 2.6.2 (CS 分解 (细形式))**　考虑矩阵

$$Q = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}, \quad Q_1 \in \mathbb{R}^{m_1 \times n}, \quad Q_2 \in \mathbb{R}^{m_2 \times n},$$

其中 $m_1 \geqslant n, m_2 \geqslant n$. 设 $Q$ 的列是规范正交的, 则存在正交矩阵 $U_1 \in \mathbb{R}^{m_1 \times m_1}$, $U_2 \in \mathbb{R}^{m_2 \times m_2}$ 和 $V_1 \in \mathbb{R}^{n \times n}$, 使得

$$\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} V_1 = \begin{bmatrix} C \\ S \end{bmatrix},$$

其中

$$\begin{aligned} C &= \mathrm{diag}(\cos(\theta_1), \cdots, \cos(\theta_n)), \\ S &= \mathrm{diag}(\sin(\theta_1), \cdots, \sin(\theta_n)), \end{aligned}$$

$$0 \leqslant \theta_1 \leqslant \theta_2 \leqslant \cdots \leqslant \theta_n \leqslant \frac{\pi}{2}.$$

**证明**　由于 $\|Q_1\|_2 \leqslant \|Q\|_2 = 1, Q_1$ 的奇异值都位于 [0, 1] 区间内, 令

$$U_1^{\mathrm{T}} Q_1 V_1 = C = \mathrm{diag}(c_1, \cdots, c_n) = \begin{bmatrix} I_t & 0 \\ 0 & \Sigma \end{bmatrix} \begin{matrix} t \\ m_1 - t \end{matrix}$$
$$\qquad\qquad\qquad\qquad\qquad\qquad\quad t \quad n-t$$

是 $Q_1$ 的 SVD, 其中

$$1 = c_1 = \cdots = c_t > c_{t+1} \geqslant \cdots \geqslant c_n \geqslant 0.$$

为完成定理之证明我们需要构造正交矩阵 $U_2$. 如果

$$Q_2 V_1 = [\ W_1 \quad W_2\ ],$$
$$\qquad\qquad t \qquad n-t$$

则

$$\begin{bmatrix} U_1 & 0 \\ 0 & I_{m_2} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} V_1 = \begin{bmatrix} I_t & 0 \\ 0 & \Sigma \\ W_1 & W_2 \end{bmatrix}.$$

由于此矩阵之列向量都是 2 范数单位向量, $W_1 = 0$. 因为

$$W_2^{\mathrm{T}} W_2 = I_{n-t} - \Sigma^{\mathrm{T}} \Sigma \equiv \mathrm{diag}(1 - c_{t+1}^2, \cdots, 1 - c_n^2)$$

非奇异, 所以 $W_2$ 的列向量非零且相互正交. 对 $k = 1 : n$ 令 $s_k = \sqrt{1 - c_k}$, 则

$$Z = W_2\ \mathrm{diag}(1/s_{t+1}, \cdots, 1/s_n)$$

的列向量是正交的. 利用定理 2.5.1 知存在正交矩阵 $U_2 \in \mathbb{R}^{m_2 \times m_2}$ 使得 $U_2(:, t+1 : n) = Z$. 容易证明

$$U_2^{\mathrm{T}} Q_2 V_1 = \mathrm{diag}(s_1, \cdots, s_n) \equiv S.$$

由于对 $1 : n$ 都有 $c_k^2 + s_k^2 = 1$, 这些量正是所需要的余弦值和正弦值. □

用同样的技巧可证明如下更一般的分解结果.

**定理 2.6.3 (CS 分解 (一般形式))** 如果

$$Q = \left[\begin{array}{c|c} Q_{11} & Q_{12} \\ \hline Q_{21} & Q_{22} \end{array}\right]$$

是 $n \times n$ 正交阵的任意 $2 \times 2$ 分划, 则存在正交阵

$$U = \left[\begin{array}{c|c} U_1 & 0 \\ \hline 0 & U_2 \end{array}\right] \quad 和 \quad V = \left[\begin{array}{c|c} V_1 & 0 \\ \hline 0 & V_2 \end{array}\right]$$

使得

$$U^{\mathrm{T}}QV=\left[\begin{array}{ccc|ccc} I & 0 & 0 & 0 & 0 & 0\\ 0 & C & 0 & 0 & S & 0\\ 0 & 0 & 0 & 0 & 0 & I\\ \hline 0 & 0 & 0 & I & 0 & 0\\ 0 & S & 0 & 0 & -C & 0\\ 0 & 0 & I & 0 & 0 & 0 \end{array}\right]$$

其中 $C=\mathrm{diag}(c_1,\cdots,c_p), S=\mathrm{diag}(s_1,\cdots,s_p)$ 且 $0<c_i,s_i<1$, $c_i^2+s_i^2=1(i=1:p)$.

**证明** 详细证明可见 Paige and Saunders(1981). 零子矩阵的维数没有给出. 这些零子矩阵有些可能是空的. □

此分解传递的重要信息是, $Q_{ij}$ 的 SVD 分解是密切相关的.

**例 2.6.1** 矩阵

$$Q=\left[\begin{array}{cc|ccc} -0.7576 & 0.3697 & 0.3838 & 0.2126 & -0.3112\\ -0.4077 & -0.1552 & -0.1129 & 0.2676 & 0.8517\\ -0.0488 & 0.7240 & -0.6730 & -0.1301 & 0.0602\\ \hline -0.2287 & 0.0088 & 0.2235 & -0.9235 & 0.2120\\ 0.4530 & 0.5612 & 0.5806 & 0.1162 & 0.3595 \end{array}\right]$$

是正交矩阵, 它在上面的划分下可化成

$$U^{\mathrm{T}}QV=\left[\begin{array}{cc|ccc} 0.9837 & 0.0000 & 0.1800 & 0.0000 & 0.0000\\ 0.0000 & 0.6781 & 0.0000 & 0.7349 & 0.0000\\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000\\ \hline 0.1800 & 0.0000 & -0.9837 & 0.0000 & 0.0000\\ 0.0000 & 0.7349 & 0.0000 & -0.6781 & 0.0000 \end{array}\right].$$

这些余弦值及正弦值所对应的角度在许多应用中是很重要的. 见 12.4 节.

## 习 题

**2.6.1** 证明: 如果 $P$ 是正交投影, 则 $Q=I-2P$ 是正交的.

**2.6.2** 正交投影的奇异值是什么?

**2.6.3** 设 $S_1=\mathrm{span}\{x\}$ 和 $S_2=\mathrm{span}\{y\}$, 其中 $x$ 和 $y$ 都是 $\mathbb{R}^2$ 中的 2 范数单位向量. 只用 $\mathrm{dist}(\cdot,\cdot)$ 的定义证明 $\mathrm{dist}(S_1,S_2)=\sqrt{1-(x^{\mathrm{T}}y)^2}$, 从而验证 $S_1$ 与 $S_2$ 的距离是 $x$ 与 $y$ 之间夹角之正弦.

## 本节注释与参考文献

下面文章讨论 CS 分解的不同性质:

C. Davis and W. Kahan(1970). "The Rotation of Eigenvectors by a Perturbation III," *SIAM J. Num. Anal.* 7, 1–46.

G. W. Stewart(1977). "On the Perurbation of Pseudo-Inverses, Projections and Linear Least Squares Problems," *SIAM Review* 19, 634–662.

C. C. Paige and M. Saunders(1981). "Toward a Generalized Singular Value Decomposition," *SIAM J. Num. Anal.* 18, 398–405.

C. C. Paige and M. Wei(1994). "History and Generality of the CS Decomposition," *Lin. Alg. and Its Applic.* 208/209, 303–326.

一些计算细节可参阅 8.7 节.

关于 CS 分解以及子空间距离的深入的几何解释可见:

T. A. Arias, A. Edelman, and S. Smith(1996). "Conjugate Gradient and Newton's method on the Grassman and Stiefel Manifolds," to appear in *SIAM J. Matrix Anal. Appl.*

## 2.7 正方形线性方程组的敏感性

现在我们用以上各节给出的工具来分析线性方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, 其中 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是非奇异的, $\boldsymbol{b} \in \mathbb{R}^n$. 我们的目的是研究对 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的扰动如何影响解 $\boldsymbol{x}$. 更详细的讨论可见 Higham(1996).

### 2.7.1 SVD 分析

如果 $\boldsymbol{A}$ 的 SVD 为

$$\boldsymbol{A} = \sum_{i=1}^{n} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathrm{T}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}},$$

则

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} = (\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}})^{-1}\boldsymbol{b} = \sum_{i=1}^{n} \frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i} v_i. \tag{2.7.1}$$

这个展开式表明, 当 $\sigma_n$ 很小时, $\boldsymbol{A}$ 或 $\boldsymbol{b}$ 的微小变化可导致 $\boldsymbol{x}$ 较大的变化.

只要我们温习定理 2.5.3 所说, $\sigma_n$ 是 $\boldsymbol{A}$ 到奇导矩阵集的距离, $\sigma_n$ 的大小会影响 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的敏感性, 这就不足为奇了. 当系数矩阵接近奇异矩阵时, 从直观上可明显看出, 解 $\boldsymbol{x}$ 将对扰动更加敏感.

### 2.7.2 条件

线性方程组敏感性的一个精确的度量可通过研究带参数方程组

$$(\boldsymbol{A} + \varepsilon\boldsymbol{F})x(\varepsilon) = \boldsymbol{b} + \varepsilon\boldsymbol{f}, \quad x(0) = \boldsymbol{x}$$

得到, 其中 $\boldsymbol{F} \in \mathbb{R}^{n\times n}$, $\boldsymbol{f} \in \mathbb{R}^n$. 如果 $\boldsymbol{A}$ 非奇异, 则很明显 $x(\varepsilon)$ 在零的一个邻域里是可微的, 而且, $\dot{x}(0) = \boldsymbol{A}^{-1}[\boldsymbol{f} - \boldsymbol{F}\boldsymbol{x}]$. 于是, $x(\varepsilon)$ 的 TayLor 级数展开具有形式

$$x(\varepsilon) = \boldsymbol{x} + \varepsilon\dot{x}(0) + O(\varepsilon^2).$$

用任一向量范数以及相应的矩阵范数可得

$$\frac{\|\boldsymbol{x}(\varepsilon)-\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leqslant |\varepsilon|\|\boldsymbol{A}^{-1}\|\left\{\frac{\|\boldsymbol{f}\|}{\|\boldsymbol{x}\|}+\|\boldsymbol{F}\|\right\}+O(\varepsilon^2). \tag{2.7.2}$$

对于方阵 $\boldsymbol{A}$, 条件数$\kappa(\boldsymbol{A})$ 定义为

$$\kappa(\boldsymbol{A})=\|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|, \tag{2.7.3}$$

当 $\boldsymbol{A}$ 奇异时 $\kappa(\boldsymbol{A})$ 为 $\infty$. 从不等式 (2.7.2) 和 $\|\boldsymbol{b}\| \leqslant \|\boldsymbol{A}\|\|\boldsymbol{x}\|$ 可得到

$$\frac{\|\boldsymbol{x}(\varepsilon)-\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leqslant \kappa(\boldsymbol{A})(\rho_A+\rho_b)+O(\varepsilon^2), \tag{2.7.4}$$

其中

$$\rho_A=|\varepsilon|\frac{\|\boldsymbol{F}\|}{\|\boldsymbol{A}\|} \quad 和 \quad \rho_b=|\varepsilon|\frac{\|\boldsymbol{f}\|}{\|\boldsymbol{b}\|}$$

分别表示 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的相对误差. 所以, $x$ 的相对误差可能是 $\kappa(\boldsymbol{A})$ 乘以 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的相对误差. 在这个意义下, 条件数 $\kappa(\boldsymbol{A})$ 量化了问题 $\boldsymbol{Ax}=\boldsymbol{b}$ 的敏感性.

注意到 $\kappa(\cdot)$ 取决于所用的范数以及所对应的下标, 例如

$$\kappa_2(\boldsymbol{A})=\|\boldsymbol{A}\|_2\|\boldsymbol{A}^{-1}\|_2=\frac{\sigma_1(\boldsymbol{A})}{\sigma_n(\boldsymbol{A})}. \tag{2.7.5}$$

因此, 矩阵 $\boldsymbol{A}$ 的 2 范数条件数度量超椭圆体 $\{\boldsymbol{Ax}:\|\boldsymbol{x}\|_2=1\}$ 的伸长.

我们给出条件数的另外两个特征. 对于 $p$ 范数条件数, 我们有

$$\frac{1}{\kappa_p(\boldsymbol{A})}=\min_{\boldsymbol{A}+\Delta\boldsymbol{A}奇异}\frac{\|\Delta\boldsymbol{A}\|_p}{\|\boldsymbol{A}\|_p}. \tag{2.7.6}$$

此结果可见于 Kahan(1996), 它表明 $\kappa_p(\boldsymbol{A})$ 度量了从 $\boldsymbol{A}$ 到奇异矩阵集之的相对 $p$ 范数距离.

对任一范数, 我们还有

$$\kappa(\boldsymbol{A})=\lim_{\varepsilon\to 0}\sup_{\|\Delta\boldsymbol{A}\|\leqslant\varepsilon\|\boldsymbol{A}\|}\frac{\|(\boldsymbol{A}+\Delta\boldsymbol{A})^{-1}-\boldsymbol{A}^{-1}\|}{\varepsilon}\frac{1}{\|\boldsymbol{A}^{-1}\|}. \tag{2.7.7}$$

此重要结果仅仅表明条件数是映射 $\boldsymbol{A}\to\boldsymbol{A}^{-1}$ 的规范化的 Frechet 导数. 更详细的讨论可见 Rice(1966b). 值得提醒的是, 我们是通过微分来导出 $\kappa(\boldsymbol{A})$ 的.

如果 $\kappa(\boldsymbol{A})$ 大, 则称 $\boldsymbol{A}$ 是病态矩阵. 注意此性质是依赖于范数的①. 但是, $\mathbb{R}^{n\times n}$ 上的任意两个条件数 $\kappa_\alpha(\cdot)$ 和 $\kappa_\beta(\cdot)$ 都是等价的, 即能找到常数 $c_1$ 和 $c_2$ 使得

$$c_1\kappa_\alpha(\boldsymbol{A}) \leqslant \kappa_\beta(\boldsymbol{A}) \leqslant c_2\kappa_\alpha(\boldsymbol{A}), \quad \boldsymbol{A}\in\mathbb{R}^{n\times n}.$$

例如, 在 $\mathbb{R}^{n\times n}$ 上我们有

① 也是依赖于"大"的定义, 这将在 3.5 节中讨论.

$$\begin{aligned} &\frac{1}{n}\kappa_2(\boldsymbol{A}) \leqslant \kappa_1(\boldsymbol{A}) \leqslant n\kappa_2(\boldsymbol{A}), \\ &\frac{1}{n}\kappa_\infty(\boldsymbol{A}) \leqslant \kappa_2(\boldsymbol{A}) \leqslant n\kappa_\infty(\boldsymbol{A}), \\ &\frac{1}{n^2}\kappa_1(\boldsymbol{A}) \leqslant \kappa_\infty(\boldsymbol{A}) \leqslant n^2\kappa_1(\boldsymbol{A}). \end{aligned} \tag{2.7.8}$$

于是, 如果一个矩阵在 $\alpha$ 范数下是病态的, 则它在 $\beta$ 范数意义下以上述常数 $c_1$ 和 $c_2$ 为权是病态的.

对任何 $p$ 范数, $\kappa_p(\boldsymbol{A}) \geqslant 1$. 条件数小的矩阵称为*良态的*. 在 2 范数意义下, 正交矩阵最良态, 因为对正交矩阵 $\boldsymbol{Q}$, 有 $\kappa_2(\boldsymbol{Q}) = 1$.

### 2.7.3 行列式与靠近奇异的程度

考虑行列式大小度量病态的好坏是很自然的. 如果 $\det(\boldsymbol{A}) = 0$ 等价于奇异, $\det(\boldsymbol{A}) \approx 0$ 是否等价于靠近奇异? 不幸的是, $\det(\boldsymbol{A})$ 和 $\boldsymbol{Ax} = \boldsymbol{b}$ 的条件数几乎没什么关系: 例如, 矩阵

$$\boldsymbol{B}_n = \begin{bmatrix} 1 & -1 & \cdots & -1 \\ 0 & 1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.7.9}$$

的行列式为 1, 但 $\kappa_\infty(\boldsymbol{A}_n) = n2^{n-1}$. 另一方面, 一个非常良态的矩阵的行列式可能很小. 例如,

$$\boldsymbol{D}_n = \mathrm{diag}(10^{-1}, \cdots, 10^{-1}) \in \mathbb{R}^{n\times n}$$

满足 $\kappa_p(\boldsymbol{D}_n) = 1$, 但 $\det(\boldsymbol{D}_n) = 10^{-n}$.

### 2.7.4 一个精确的范数界

重温 (2.7.4) 之推导是有价值的, 因为它揭露了 $\kappa(\boldsymbol{A})$ 与 $x(\varepsilon)$ 在 $\varepsilon = 0$ 处的变化率之间的关系. 但是, 稍微不足的是它基于 $\varepsilon$ 是"足够小", 而且对 $O(\varepsilon^2)$ 项之大小没有阐明. 在本节与下一节我们讨论一些非常精确的 $\boldsymbol{Ax} = \boldsymbol{b}$ 之扰动定理.

首先, 我们证明一有用的引理, 它用 $\kappa(\boldsymbol{A})$ 表明什么情况下一个扰动方程组仍是非奇异的.

**引理 2.7.1** 假设

$$\begin{aligned} &\boldsymbol{Ax} = \boldsymbol{b}, && \boldsymbol{A} \in \mathbb{R}^{n\times n}, && \boldsymbol{0} \neq \boldsymbol{b} \in \mathbb{R}^n, \\ &(\boldsymbol{A} + \Delta\boldsymbol{A})\boldsymbol{y} = \boldsymbol{b} + \Delta\boldsymbol{b}, && \Delta\boldsymbol{A} \in \mathbb{R}^{n\times n}, && \Delta\boldsymbol{b} \in \mathbb{R}^n, \end{aligned}$$

其中 $\|\Delta\boldsymbol{A}\| \leqslant \varepsilon\|\boldsymbol{A}\|$ 和 $\|\Delta\boldsymbol{b}\| \leqslant \varepsilon\|\boldsymbol{b}\|$. 如果 $\varepsilon\kappa(\boldsymbol{A}) = r < 1$, 则 $\boldsymbol{A} + \Delta\boldsymbol{A}$ 非奇异且

$$\frac{\|\boldsymbol{y}\|}{\|\boldsymbol{x}\|} \leqslant \frac{1+r}{1-r}.$$

**证明**　由于 $\|A^{-1}\Delta A\|_r \leqslant \varepsilon\|A^{-1}\|\|A\| = r < 1$, 从定理 2.3.4 知 $(A+\Delta A)$ 非奇异. 利用引理 2.3.3 和等式 $(I + A^{-1}\Delta A)y = x + A^{-1}\Delta b$, 我们发现

$$\begin{aligned}\|y\| &\leqslant \|(I + A^{-1}\Delta A)^{-1}\|(\|x\| + \varepsilon\|A^{-1}\|\|b\|) \\ &\leqslant \frac{1}{1-r}(\|x\| + \varepsilon\|A^{-1}\|\|b\|) = \frac{1}{1-r}\left(\|x\| + r\frac{\|b\|}{\|A\|}\right).\end{aligned}$$

因为 $\|b\| = \|Ax\| \leqslant \|A\|\|x\|$, 故知

$$\|y\| \leqslant \frac{1}{1-r}(\|x\| + r\|x\|).$$ □

现在我们给出 $Ax = b$ 的一个精确的扰动界.

**定理 2.7.2**　如果引理 2.7.1 中的条件满足, 则

$$\frac{\|y - x\|}{\|x\|} \leqslant \frac{2\varepsilon}{1-r}\kappa(A). \tag{2.7.10}$$

**证明**　由于

$$y - x = A^{-1}\Delta b - A^{-1}\Delta A y, \tag{2.7.11}$$

我们有 $\|y - x\| \leqslant \varepsilon\|A^{-1}\|\|b\| + \varepsilon\|A^{-1}\|\|A\|\|y\|$, 于是

$$\begin{aligned}\frac{\|y - x\|}{\|x\|} &\leqslant \varepsilon\kappa(A)\frac{\|b\|}{\|A\|\|x\|} + \varepsilon\kappa(A)\frac{\|y\|}{\|x\|} \\ &\leqslant \varepsilon\kappa(A)\left(1 + \frac{1+r}{1-r}\right) = \frac{2\varepsilon}{1-r}\kappa(A).\end{aligned}$$ □

**例 2.7.1**　问题 $Ax = b$

$$\begin{bmatrix} 1 & 0 \\ 0 & 10^{-6} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 10^{-6} \end{bmatrix}$$

之解为 $x = (1,1)^{\mathrm{T}}$ 且条件数 $\kappa_\infty(A) = 10^6$. 如果 $\Delta b = (10^{-6}, 0)^{\mathrm{T}}$, $\Delta A = 0$ 且 $(A + \Delta A)y = b + \Delta b$, 则 $y = (1 + 10^{-6}, 1)^{\mathrm{T}}$, 此时不等式 (2.7.10) 为

$$10^{-6} = \frac{\|x - y\|_\infty}{\|x\|_\infty} \ll \frac{\|\Delta b\|_\infty}{\|b\|_\infty}\kappa_\infty(A) = 10^{-6}10^6 = 1.$$

因此, (2.7.10) 的上界可能是扰动所导致的误差之非常粗的过高估计. 另一方面, 如果 $\Delta b = (10, 10^{-6})^{\mathrm{T}}$, $\Delta A = 0$ 且 $(A + \Delta A)y = b + \Delta b$, 则不等式 (2.7.10) 为

$$\frac{10^0}{10^0} \leqslant 2 \times 10^{-6}10^6,$$

所以, 有扰动使 (2.7.10) 的界基本上可以达到.

### 2.7.5 一些关于分量的精确界

我们以证明对于分量扰动界存在更精细的扰动理论来结束本节. 这需要利用绝对值记号.

**定理 2.7.3** 设

$$\begin{aligned} \boldsymbol{Ax} &= \boldsymbol{b}, & \boldsymbol{A} &\in \mathbb{R}^{n\times n}, & \boldsymbol{0} &\neq \boldsymbol{b} \in \mathbb{R}^n, \\ (\boldsymbol{A}+\Delta\boldsymbol{A})\boldsymbol{y} &= \boldsymbol{b}+\Delta\boldsymbol{b}, & \Delta\boldsymbol{A} &\in \mathbb{R}^{n\times n}, & \Delta\boldsymbol{b} &\in \mathbb{R}^n, \end{aligned}$$

以及 $\|\Delta\boldsymbol{A}\| \leqslant \varepsilon\|\boldsymbol{A}\|$ 和 $\|\Delta\boldsymbol{b}\| \leqslant \varepsilon\|\boldsymbol{b}\|$. 如果 $\varepsilon\kappa_\infty(\boldsymbol{A}) = r < 1$, 则 $(\boldsymbol{A}+\Delta\boldsymbol{A})$ 非奇异且

$$\frac{\|\boldsymbol{y}-\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} \leqslant \frac{2\varepsilon}{1-r}\||\boldsymbol{A}^{-1}||\boldsymbol{A}|\|_\infty.$$

**证明** 由于 $\|\Delta\boldsymbol{A}\|_\infty \leqslant \varepsilon\|\boldsymbol{A}\|_\infty$ 和 $\|\Delta\boldsymbol{b}\|_\infty \leqslant \varepsilon\|\boldsymbol{b}\|_\infty$, 引理 2.7.1 的条件在 $\infty$ 范数下是满足的. 于是 $\boldsymbol{A}+\Delta\boldsymbol{A}$ 非奇异且

$$\frac{\|\boldsymbol{y}\|_\infty}{\|\boldsymbol{x}\|_\infty} \leqslant \frac{1+r}{1-r}.$$

利用 (2.7.11) 我们发现

$$\begin{aligned} |\boldsymbol{y}-\boldsymbol{x}| &\leqslant |\boldsymbol{A}^{-1}||\Delta\boldsymbol{b}| + |\boldsymbol{A}^{-1}||\Delta\boldsymbol{A}||\boldsymbol{y}| \\ &\leqslant \varepsilon|\boldsymbol{A}^{-1}||\boldsymbol{b}| + \varepsilon|\boldsymbol{A}^{-1}||\boldsymbol{A}||\boldsymbol{y}| \leqslant \varepsilon|\boldsymbol{A}^{-1}||\boldsymbol{A}|(|\boldsymbol{x}|+|\boldsymbol{y}|). \end{aligned}$$

两边取范数, 则有

$$\|\boldsymbol{y}-\boldsymbol{x}\|_\infty \leqslant \varepsilon\||\boldsymbol{A}^{-1}||\boldsymbol{A}|\|_\infty\left(\|\boldsymbol{x}\|_\infty + \frac{1+r}{1-r}\|\boldsymbol{x}\|_\infty\right).$$

不等式两边同除以 $\|\boldsymbol{x}\|_\infty$ 即得到定理. □

我们把量 $\||\boldsymbol{A}^{-1}||\boldsymbol{A}|\|_\infty$ 称为 Skeel条件数. 它对几个重要的线性方程组计算的分析非常有用. 见 3.5 节.

最后, 我们介绍 Oettli and Prager(1964) 的结果, 它指明什么情形下 $n\times n$ 方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的一个近似解 $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ 满足给定结构的扰动方程组. 特别地, 假设 $\boldsymbol{E}\in\mathbb{R}^{n\times n}$ 和 $\boldsymbol{f}\in\mathbb{R}^n$ 给定且它们的元素非负. 我们寻找 $\Delta\boldsymbol{A}\in\mathbb{R}^{n\times n}$, $\Delta\boldsymbol{b}\in\mathbb{R}^n$ 和 $w>0$ 使得

$$(\boldsymbol{A}+\Delta\boldsymbol{A})\hat{\boldsymbol{x}} = \boldsymbol{b}+\Delta\boldsymbol{b}, \quad |\Delta\boldsymbol{A}| \leqslant w\boldsymbol{E}, \quad |\Delta\boldsymbol{b}| \leqslant w\boldsymbol{f}. \tag{2.7.12}$$

注意, 适当地选取 $\boldsymbol{E}$ 和 $\boldsymbol{f}$, 从扰动的方程组可得某特定量. 例如, 当 $\boldsymbol{E}=|\boldsymbol{A}|$ 和 $\boldsymbol{f}=|\boldsymbol{b}|$ 以及 $w$ 很小时, $\hat{\boldsymbol{x}}$ 满足一个附近的方程组 (在向量分量的意义下). Oettli and Prager(1964) 证明了, 对给定 $\boldsymbol{A}$, $\boldsymbol{b}$, $\hat{\boldsymbol{x}}$, $\boldsymbol{E}$ 和 $\boldsymbol{f}$, 在 (2.7.12) 中, 可能的最小的 $w$ 是

$$w_{\min} = \max_{1\leqslant i\leqslant n} \frac{|A\hat{x} - b|_i}{(E|\hat{x}| + f)_i}.$$

如果 $A\hat{x} = b$ 则 $w_{\min} = 0$. 在另一极端情况, 如果 $w_{\min} = \infty$, 则 $\hat{x}$ 不满足给定的扰动结构之任何方程组.

## 习 题

**2.7.1** 证明: 如果 $\|I\| \geqslant 1$, 则 $\kappa(A) \geqslant 1$.

**2.7.2** 证明: 对任何给定范数, $\kappa(AB) \leqslant \kappa(A)\kappa(B)$ 且对任何正数 $\alpha$ 有 $\kappa(\alpha A) = \kappa(A)$.

**2.7.3** 给出 $X \in \mathbb{R}^{m\times n} (m \geqslant n)$ 的 2 范数条件数与矩阵

$$B = \begin{bmatrix} I_m & X \\ 0 & I_n \end{bmatrix} \quad \text{和} \quad C = \begin{bmatrix} X \\ I_n \end{bmatrix}$$

的 2 范数条件数之间的关系.

## 本节注释与参考文献

条件数在下面文章中有透彻的讨论:

J. Rice(1966). "A Theory of Condition," *SIAM J. Num. Anal.* 3, 287–310.

W. Kahan(1966). "Numerical Linear Algebra," *Canadian Math. Bull.* 9, 757–801.

关于以分量形式的扰动理论的参考文献包括:

W. Oettli and W. Prager(1964). "Compatibility of Approximate Solutions of Linear Equations with Given Error Bounds for Coefficients and Right Hand Sides," *Numer. Math.* 6, 405–409.

J. E. Cope and B. W. Rust(1979). "Bounds on solutions of systems with accurate data," *SIAMJ. Num. Anal.* 16, 950–963.

R. D. Skeel(1979). "Scaling for numerical stability in Gaussian Elimination," *J. ACM* 26, 494–526.

J. W. Demmel(1992). "The Componentwise Distance to the Nearest Singular Matrix," *SIAMJ. Matrix Anal. Appl.* 13, 10–19.

D. J. Higham and N. J. Higham(1992). "Componentwise Perturbation Theory for Linear Systems with Multiple Right-Hand Sides," *Lin. Alg. and Its Applic.* 174, 111–129.

N. J. Higham(1994). "A Survey of Componentwise Perturbation Theory in Numerical Linear Algebra," in *Mathematics of Computation* 1943–1993: *A Half Century of Computational Mathematics*, W.Gautschi(ed.), Volume 48 of *Proceedings of Symposia in Applied Mathematics*, American Mathematical Society, Providence, Rhode Island.

S. Chandrasekaren and I. C. F. Ipsen(1995). "On the Sensitivity of Solution Components in Linear Systems of Equations," *SIAM J. Matrix Anal. Appl.* 16, 93–112.

条件数之倒数给出了 $Ax = b$ 靠近奇异的程度的度量, 知道一个给定问题离一个难的或者是不可解问题的靠近程度之重要性在许多计算中是很有帮助的, 见:

A. Laub(1985). "Numerical Linear Algebra Aspects of Control Design Computations," *IEEE Trans. Auto. Cont.* *AC*-30, 97–108.

J. L. Barlow(1986). "On the Smallest Positive Singular Value of an $M$-Matrix with Applications to Ergodic Markov Chains," *SIAM. J Alg. and Disc. Struct.* 7, 414–424.

J. W. Demmel(1987). "On the Distance to the Nearest Ill-Posed Problem," *Numer. Math.* 51, 251–289.

J. W. Demmel(1988). "The Probability that a Numerical Analysis Problem is Difficult," *Math. Comp.* 50, 449–480.

N. J. Higham(1989). "Matrix Nearness Problems and Applications," in *Applications of Matrix Theory*, M. J. C. Gover and S. Barnett(eds), Oxford University Press, Oxford UK,1–27.

# 第3章　一般线性方程组

求解线性方程组 $Ax = b$ 是科学计算的中心问题. 本章我们集中讨论高斯消去法, 这是处理 $A$ 是方的、稠密的以及无结构时的首选算法. 如果 $A$ 不是此类矩阵, 相关的算法可见于第 4, 5, 10 章. $Ax = b$ 的一些并行求解方法在第 6 章中讨论.

在 3.1 节我们通过讨论求解三角方程组之容易来导出高斯消去法. 然后 3.2 节阐述通过高斯变换将一般方程组转化为三角方程组, 为此引进了矩阵分解的语言. 不幸的是, 得到的方法对一类非平凡的问题表现很差. 3.3 节的误差分析指出了困难所在并引出了 3.4 节, 在那里给出了选主元的概念. 在最后一节我们详述了关于加权、迭代改进以及条件数估计的一些重要的实际问题.

*预备知识*

阅读本章还需要掌握第 1 章、2.1~2.5 节和 2.7 节的知识. 其他的参考文献包括 Forsythe and Moler(1967), Stewart(1973), Hager(1983), Watkins(1991), Ciarlet(1992), Datta (1995), Higham(1996), Trefethan and Bau(1996) 以及 Demmel(1996). 对本章很重要的一些 MATLAB 函数有 `lu,cond,rcond` 以及反斜线算子 "\". 与 LAPACK 的相关程序为

| LAPACK: 三角方程组 | |
|---|---|
| _TRSV | 解 $Ax = b$ |
| _TRSM | 解 $AX = B$ |
| _TRCON | 条件数估计 |
| _TRRFS | 解 $AX = B, A^TX = B$, 给出误差界 |
| _TRTRS | 解 $AX = B, A^TX = B$ |
| _TRTRI | $A^{-1}$ |

| LAPACK: 一般线性方程组 | |
|---|---|
| _GESV | 解 $AX = B$ |
| _GECON | 通过 $PA = LU$ 估计条件数 |
| _GERFS | 改进 $AX = B, A^TX = B, A^HX = B$ 之解, 给出误差界 |
| _GESVX | 解 $AX = B, A^TX = B, A^HX = B$ |
| _GETRF | $PA = LU$ |
| _GETRS | 利用 $PA = LU$ 解 $AX = B, A^TX = B, A^HX = B$ |
| _GETRI | $A^{-1}$ |
| _GEEQU | 平衡方程 |

## 3.1　三角方程组

传统的线性方程组的分解方法涉及将给定的正方线性方程组转化为具有同样解的三角方程组. 本节就是关于三角方程组之求解.

### 3.1.1 向前消去法

考虑如下 $2\times 2$ 下三角方程组

$$\begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

如果 $l_{11}l_{22} \neq 0$, 则未知数可依次确定:

$$x_1 = b_1/l_{11},$$
$$x_2 = (b_2 - l_{21}x_1)/l_{22}.$$

这就是称之为*向前消去法*的算法之 $2\times 2$ 形式. 通过解 $\boldsymbol{Lx}=\boldsymbol{b}$ 的第 $i$ 个方程求出 $x_i$ 即可得到此算法的一般形式

$$x_i = \left(b_i - \sum_{j=1}^{i-1} l_{ij}x_j\right) \Big/ l_{ii}.$$

如果对 $i=1:n$ 计算上式, 则 $\boldsymbol{x}$ 的所有分量都可求得. 注意到在第 $i$ 步需要计算 $L(i,1:i-1)$ 与 $x(1:i-1)$ 的点积. 由于 $b_i$ 仅在计算 $x_i$ 的公式中用到, 前者可被后者覆盖.

**算法 3.1.1 (向前消去法: 行形式)** 设 $\boldsymbol{L}\in\mathbb{R}^{n\times n}$ 是下三角形矩阵, $\boldsymbol{b}\in\mathbb{R}^n$, 则此算法用 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解覆盖 $\boldsymbol{b}$. 假定 $\boldsymbol{L}$ 是非奇异的.

$b(1) = b(1)/L(1,1)$
**for** $i = 2:n$
  $b(i) = (b(i) - L(i,1:i-1)b(1:i-1))/L(i,i)$
**end**

此算法需要 $n^2$ 个 flop, 注意到 $\boldsymbol{L}$ 是按行调用的. 计算出来的解 $\hat{\boldsymbol{x}}$ 满足

$$(\boldsymbol{L}+\boldsymbol{F})\hat{\boldsymbol{x}} = \boldsymbol{b}, \quad |\boldsymbol{F}| \leqslant n\mathbf{u}|\boldsymbol{L}| + O(\mathbf{u}^2). \tag{3.1.1}$$

其证明可见 Higham(1996). 这说明计算的解精确地满足一个小扰动的方程. 而且, 扰动矩阵 $\boldsymbol{F}$ 的每一个元素相对于所对应的元素来说都是小的.

### 3.1.2 向后消去法

解上三角方程组 $\boldsymbol{Ux}=\boldsymbol{b}$ 的类似算法叫*向后消去法*. $x_i$ 的计算公式为

$$x_i = \left(b_i - \sum_{j=i+1}^{n} u_{ij}x_j\right) \Big/ u_{ii},$$

同样 $b_i$ 可以被 $x_i$ 所覆盖.

**算法 3.1.2 (向后消去法: 行形式)** 如果 $\boldsymbol{U}\in\mathbb{R}^{n\times n}$ 是上三角形矩阵, $\boldsymbol{b}\in\mathbb{R}^n$, 则此算法用 $\boldsymbol{Ux}=\boldsymbol{b}$ 之解覆盖 $\boldsymbol{b}$. 假定 $\boldsymbol{U}$ 是非奇异的.

$b(n) = b(n)/U(n,n)$

```
for i = n-1 : -1 : 1
    b(i) = (b(i) - U(i, i+1:n)b(i+1:n))/U(i,i)
end
```

此算法需要 $n^2$ 个 flop, $U$ 是按行调用的, 可证明计算出来的 $\hat{x}$ 满足

$$(\boldsymbol{U}+\boldsymbol{F})\hat{\boldsymbol{x}}=\boldsymbol{b}, \quad |\boldsymbol{F}| \leqslant n\mathbf{u}|U| + O(\mathbf{u}^2). \tag{3.1.2}$$

### 3.1.3 基于列的形式

交换循环顺序可得到以上算法的基于列的形式. 为了从代数的角度理解它, 我们考虑向前消去法, 一旦 $x_1$ 被解出来, 该变量可从第 2 个至第 $n$ 个方程中去掉, 我们可只考虑缩小后的方程组 $L(2:n,2:n)x(2:n)=b(2:n)-x(1)L(2:n,1)$. 然后, 我们算出 $x_2$. 并且从第 3 个至第 $n$ 个方程中去掉 $x_2$, 依次类推. 于是, 如果此方法用于

$$\begin{bmatrix} 2 & 0 & 0 \\ 1 & 5 & 0 \\ 7 & 9 & 8 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 2 \\ 5 \end{bmatrix},$$

则我们有 $x_1=3$, 然后我们处理 $2\times 2$ 方程组

$$\begin{bmatrix} 5 & 0 \\ 9 & 8 \end{bmatrix}\begin{bmatrix} x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix} - 3\begin{bmatrix} 1 \\ 7 \end{bmatrix} = \begin{bmatrix} -1 \\ -16 \end{bmatrix}.$$

以下是利用覆盖的完整过程.

**算法 3.1.3 (向前消去法: 列形式)** *设 $L \in \mathbb{R}^{n\times n}$ 是下三角形矩阵, $\boldsymbol{b}\in\mathbb{R}^n$, 则此算法用 $L\boldsymbol{x}=\boldsymbol{b}$ 之解覆盖 $b$. 假设 $L$ 是非奇异的.*

```
for j = 1 : n-1
    b(j) = b(j)/L(j,j)
    b(j+1:n) = b(j+1:n) - b(j)L(j+1:n, j)
end
b(n) = b(n)/L(n,n)
```

同样, 可以得到基于列的 saxpy 算法来进行向后消去.

**算法 3.1.4 (向后消去法: 行形式)** *设 $U \in \mathbb{R}^{n\times n}$ 是上三角形矩阵, $\boldsymbol{b}\in\mathbb{R}^n$, 则此算法用 $U\boldsymbol{x}=\boldsymbol{b}$ 之解覆盖 $b$. 假设 $U$ 是非奇异的.*

```
for j = n : -1 : 2
    b(j) = b(j)/U(j,j)
    b(1:j-1) = b(1:j-1) - b(j)U(1:j-1, j)
end
b(1) = b(1)/U(1,1)
```

注意到算法 3.1.3 和算法 3.1.4 中的主要运算是 saxpy 运算. 这些 saxpy 型算法的舍入误差表现与点积型算法是基本相同的.

三角方程组的计算解的精度常常是惊人的好. 见 Higham(1996).

### 3.1.4 多右端项问题

考虑计算 $\boldsymbol{LX}=\boldsymbol{B}$ 的 $\boldsymbol{X}\in\mathbb{R}^{n\times q}$ 的问题, 其中 $\boldsymbol{L}\in\mathbb{R}^{n\times n}$ 是下三角形矩阵, $\boldsymbol{B}\in\mathbb{R}^{n\times q}$. 这就是*多右端项*的向前消去问题. 我们证明此问题可用一分块算法求解. 分块算法在 $q$ 和 $n$ 足够大的矩阵乘法中是经常用到的. 这一点在以下几节中讨论各种各样的分块分解算法中是重要的. 我们指出, 虽然这里考虑的是下三角问题, 但所有我们提到的内容都可用于上三角情形.

为得到分块向前消去法, 我们把方程 $\boldsymbol{LX}=\boldsymbol{B}$ 划分如下:

$$\begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{L}_{N1} & \boldsymbol{L}_{N2} & \cdots & \boldsymbol{L}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \\ \vdots \\ \boldsymbol{B}_N \end{bmatrix}. \tag{3.1.3}$$

假定对角块是方的. 与算法 3.1.3 的构造类似, 我们从 $\boldsymbol{L}_{11}\boldsymbol{X}_1=\boldsymbol{B}_1$ 解出 $\boldsymbol{X}_1$, 然后从第 2 块至第 $N$ 块方程中消去 $\boldsymbol{X}_1$:

$$\begin{bmatrix} \boldsymbol{L}_{22} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{L}_{32} & \boldsymbol{L}_{33} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{L}_{N2} & \boldsymbol{L}_{N3} & \cdots & \boldsymbol{L}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{X}_2 \\ \boldsymbol{X}_3 \\ \vdots \\ \boldsymbol{X}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}_2-\boldsymbol{L}_{21}\boldsymbol{X}_1 \\ \boldsymbol{B}_3-\boldsymbol{L}_{31}\boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{B}_N-\boldsymbol{L}_{N1}\boldsymbol{X}_1 \end{bmatrix}.$$

依次类推我们得到如下分块 saxpy 向前消去法:

```
for j = 1 : N
    Solve L_jj X_j = B_j
    for i = j + 1 : N
        B_i = B_i - L_ij X_j                (3.1.4)
    end
end
```

注意到在 $i$ 循环中只有一个分块 saxpy 修正:

$$\begin{bmatrix} \boldsymbol{B}_{j+1} \\ \vdots \\ \boldsymbol{B}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}_{j+1} \\ \vdots \\ \boldsymbol{B}_N \end{bmatrix} - \begin{bmatrix} \boldsymbol{L}_{j+1,j} \\ \vdots \\ \boldsymbol{L}_{N,j} \end{bmatrix} \boldsymbol{X}_j.$$

为了使此计算在一给定结构下作为矩阵相乘处理, 很明显 (3.1.3) 中的划分必须有足够 “大” 的 $\boldsymbol{X}_j$. 我们假定如果, 则每个 $\boldsymbol{X}_j$ 至少有 $r$ 行正是这种情形. 此时, 可令 $N=\text{ceil}(n/r), \boldsymbol{X}_1,\cdots \boldsymbol{X}_{N-1}\in\mathbb{R}^{r\times q}$ 且 $\boldsymbol{X}_N\in\mathbb{R}^{(n-(N-1)r)\times q}$.

### 3.1.5 3 级比例

在一个给定的算法中采用一个度量矩阵乘法的量是便利的. 对此, 我们把矩阵乘法中 flop 所占比例的定义为一个算法的 3 *级比例*. 矩阵乘法的 flop 称为 3 *级* flop.

在假定 $n = rN$ 时考查 (3.1.4) 的 3 级比例. (同样的结论对上述非均匀分块也成立.) 由于有 $N$ 个 $r \times r$ 向前消去 (计算中的 2 级的部分), 而所有的 flop 数为 $n^2$, 故知 3 级比例大约是

$$1 - \frac{Nr^2}{n^2} = 1 - \frac{1}{N}.$$

因而, 对很大的 $N$ 几乎所有的 flop 都是 3 级 flop. 只要所用的计算机在处理长度至少为 $r = n/N$ 的分块 saxpy 时能达到高性能, 就应该把 $N$ 取得尽可能大.

### 3.1.6 求解非方的三角方程组

求解非方 $m \times n$ 三角方程组问题也值得讨论. 首先考虑当 $m \geqslant n$ 时的下三角方程组, 即

$$\begin{bmatrix} \boldsymbol{L}_{11} \\ \boldsymbol{L}_{21} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}, \qquad \begin{matrix} \boldsymbol{L}_{11} \in \mathbb{R}^{n \times n}, & \boldsymbol{b}_1 \in \mathbb{R}^n, \\ \boldsymbol{L}_{21} \in \mathbb{R}^{(m-n) \times n}, & \boldsymbol{b}_2 \in \mathbb{R}^{m-n}, \end{matrix}$$

假定 $\boldsymbol{L}_{11}$ 是下三角形矩阵且非奇异. 如果我们将向前消去法用于 $\boldsymbol{L}_{11}\boldsymbol{x} = \boldsymbol{b}_1$, 则知只要 $\boldsymbol{L}_{21}(\boldsymbol{L}_{11}^{-1}\boldsymbol{b}_1) = \boldsymbol{b}_2, \boldsymbol{x}$ 就是方程组之解. 否则, 整个方程组无解. 在此情形下, 或许最小二乘解是合适的. 见第 5 章.

现在考虑在列数 $n$ 大于行数 $m$ 时的下三角方程组 $\boldsymbol{Lx} = \boldsymbol{b}$. 在此情形利用向前消去法求解 $L(1:m, 1:m)x(1:m) = \boldsymbol{b}$ 可得 $x(1:m)$, 但 $x(m+1:n)$ 是任意的, 关于未知量个数大于方程个数的更多的讨论见 5.7 节.

处理非方的上三角方程组是类似的, 详细的讨论留给读者.

### 3.1.7 单位三角方程组

单位三角形矩阵是指对角线元素全为 1 的三角形矩阵. 下面要讨论的许多三角形矩阵计算都有此附加性质. 很明显, 对以上的算法它不会带来任何困难.

### 3.1.8 三角形矩阵的代数

为以后参考, 我们列出三角形矩阵和单位三角形矩阵之乘积以及逆的一些性质.

- *上 (下) 三角形矩阵的逆是上 (下) 三角形矩阵.*
- *两个上 (下) 三角形矩阵之积是上 (下) 三角形矩阵.*
- *单位上 (下) 三角形矩阵之逆是单位上 (下) 三角形矩阵.*
- *两个单位上 (下) 三角形矩阵之积是单位上 (下) 三角形矩阵.*

## 习 题

**3.1.1** 给出一个算法计算非零向量 $\boldsymbol{z}\in\mathbb{R}^n$, 使得 $\boldsymbol{Uz}=\boldsymbol{0}$, 其中 $\boldsymbol{U}\in\mathbb{R}^{n\times n}$ 是上三角形矩阵且有 $u_{nn}=0, u_{11}\cdots u_{n-1,n-1}\neq 0$.

**3.1.2** 讨论如何计算方的三角形矩阵之行列式且能尽可能小发生上溢和下溢.

**3.1.3** 改写算法 3.1.4, 假定 $\boldsymbol{U}$ 是按列贮存于长度为 $n(n+1)/2$ 的数组 u.vec 中.

**3.1.4** 写出 (3.1.4) 的详细形式. 不假定 $N$ 整除 $n$.

**3.1.5** 证明 3.1.8 节中列出的三角形矩阵之性质.

**3.1.6** 设 $\boldsymbol{S},\boldsymbol{T}\in\mathbb{R}^{n\times n}$ 是上三角形矩阵且 $(\boldsymbol{ST}-\lambda\boldsymbol{I})\boldsymbol{x}=\boldsymbol{b}$ 是非奇异方程组. 给出一个计算 $\boldsymbol{x}$ 的 $O(n^2)$ 算法. 注意 $\boldsymbol{ST}-\lambda\boldsymbol{I}$ 的显式公式需要 $O(n^3)$ 个 flop. 提示:

$$\boldsymbol{S}_+=\begin{bmatrix}\sigma & u^{\mathrm{T}}\\ 0 & \boldsymbol{S}_c\end{bmatrix},\quad \boldsymbol{T}_+=\begin{bmatrix}\tau & v^{\mathrm{T}}\\ 0 & \boldsymbol{T}_c\end{bmatrix},\quad \boldsymbol{b}_+=\begin{bmatrix}\beta\\ b_c\end{bmatrix},$$

其中 $\boldsymbol{S}_+=S(k-1:n,k-1:n), \boldsymbol{T}_+=T(k-1:n,k-1:n), \boldsymbol{b}_+=b(k-1:n), \sigma,\tau,\beta\in\mathbb{R}$. 证明: 如果 $\boldsymbol{x}_c$ 满足

$$(\boldsymbol{S}_c\boldsymbol{T}_c-\lambda\boldsymbol{I})\boldsymbol{x}_c=\boldsymbol{b}_c,$$

以及 $\boldsymbol{w}_c=\boldsymbol{T}_c\boldsymbol{x}_c$, 则

$$\boldsymbol{x}_+=\begin{bmatrix}\gamma\\ \boldsymbol{x}_c\end{bmatrix},\quad \gamma=\frac{\beta-\sigma v^{\mathrm{T}}\boldsymbol{x}_c-u^{\mathrm{T}}w_c}{\sigma\tau-\lambda}$$

是 $(\boldsymbol{S}_+\boldsymbol{T}_+-\lambda\boldsymbol{I})\boldsymbol{x}_+=\boldsymbol{b}_+$ 的解. 注意到 $\boldsymbol{x}_+$ 和 $\boldsymbol{w}_+=\boldsymbol{T}_+\boldsymbol{x}_+$ 都需要 $O(n-k)$ 个 flop.

**3.1.7** 假定 $\boldsymbol{R}_1,\cdots,\boldsymbol{R}_p\in\mathbb{R}^{n\times n}$ 都是上三角形矩阵. 给出一个求解方程组 $(\mathbf{R}_1\cdots\mathbf{R}_P-\lambda\boldsymbol{I})\boldsymbol{x}=\boldsymbol{b}$ 的 $O(pn^2)$ 算法, 假定系数矩阵是非奇异的. 提示: 推广上一问题的答案.

## 本节注释与参考文献

三角方程组的精度之分析可见:

N. J. Higham(1989). "The Accuracy of Solutions to Triangular Systems", *SIAM J. Num. Anal.* 26. 1252–1265.

# 3.2 LU 分 解

正如我们刚才所见, 三角方程组很 "容易" 求解. 高斯消去法的思想是将一给定方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 转化为等价的三角方程组. 该转化可通过方程的适当线性组合来实现. 例如, 在方程组

$$\begin{cases}3x_1+5x_2=9\\ 6x_1+7x_2=4\end{cases}$$

中, 我们将第一个方程乘以 2, 并且在第 2 个方程中减去它则得到

$$\begin{cases}3x_1+5x_2=9\\ \qquad -3x_2=-14,\end{cases}$$

这就是 $n=2$ 的高斯消去法. 本节的目的就是对这一核心算法给出完整的叙述且用矩阵分解的语言来刻画它. 这意味着此算法求出一个单位下三角形矩阵 $\boldsymbol{L}$ 和一个上三角形矩阵 $\boldsymbol{U}$ 使得 $\boldsymbol{A}=\boldsymbol{LU}$, 即

$$\begin{bmatrix} 3 & 5 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} 3 & 5 \\ 0 & -3 \end{bmatrix}.$$

然后, 原始问题 $\boldsymbol{Ax}=\boldsymbol{b}$ 之解可通过两个三角求解过程来得到:

$$\boldsymbol{Ly}=\boldsymbol{b},\quad \boldsymbol{Ux}=\boldsymbol{y} \Rightarrow \boldsymbol{Ax}=\boldsymbol{LUx}=\boldsymbol{Ly}=\boldsymbol{b}.$$

LU 分解是高斯消去法的 "高级" 代数描述. 把一个矩阵算法用矩阵分解的 "语言" 来表达是很值得的. 它有助于推广到一般情形而且能揭露算法间的关系, 这些算法从标量层次上看可能是非常不同的.

### 3.2.1 高斯变换

为得到高斯消去法的分解描述, 我们需要消零过程的矩阵描述. 在 $n=2$ 的水平上, 如果 $x_1\neq 0$ 且 $\tau=x_2/x_1$, 则

$$\begin{bmatrix} 1 & 0 \\ -\tau & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ 0 \end{bmatrix}.$$

更一般地, 设 $\boldsymbol{x}\in\mathbb{R}^n$ 且 $x_k\neq 0$, 如果

$$\boldsymbol{\tau}^{\mathrm{T}}=(\underbrace{0,\cdots,0}_{k},\tau_{k+1},\cdots,\tau_n),\quad \tau_i=\frac{x_i}{x_k},\quad i=k+1:n,$$

且我们定义

$$\boldsymbol{M}_k=\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_k^{\mathrm{T}}, \tag{3.2.1}$$

则

$$\boldsymbol{M}_k\boldsymbol{x}=\begin{bmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & 1 & 0 & & 0 \\ 0 & & -\tau_{k+1} & 1 & & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & -\tau_n & 0 & \cdots & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ \vdots \\ x_k \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

一般地, 形如 $\boldsymbol{M}_k=\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_k^{\mathrm{T}}\in\mathbb{R}^{n\times n}$ 的矩阵在 $\boldsymbol{\tau}\in\mathbb{R}^n$ 的前 $k$ 个分量都为零时一般称为**高斯变换**. 这样的矩阵是单位下三角形矩阵. $\tau(k+1:n)$ 的元素称为**乘子**. 向量 $\boldsymbol{\tau}$ 称为**高斯向量**.

### 3.2.2 作用高斯变换

用高斯变换相乘特别简单. 设 $\boldsymbol{C}\in\mathbb{R}^{n\times r}$ 和 $\boldsymbol{M}_k=\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_k$ 是高斯变换, 则

$$\boldsymbol{M}_k\boldsymbol{C}=(\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_k^{\mathrm{T}})\boldsymbol{C}=\boldsymbol{C}-\boldsymbol{\tau}(\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{C})=\boldsymbol{C}-\boldsymbol{\tau}C(k,:).$$

是一个外积修正. 由于 $\tau(1:k)=0$, 故只有 $C(k+1:n,:)$ 受到影响, 修正 $\boldsymbol{C}=\boldsymbol{M}_k\boldsymbol{C}$ 依行计算如下:

$$
\begin{aligned}
&\textbf{for } i=k+1:n\\
&\qquad C(i,:)=C(i,:)-\tau_i C(k,:)\\
&\textbf{end}
\end{aligned}
$$

此计算需要 $2(n-1)r$ 个 flop.

**例 3.2.1**

$$
\boldsymbol{C}=\begin{bmatrix}1&4&7\\2&5&8\\3&6&10\end{bmatrix},\quad \boldsymbol{\tau}=\begin{bmatrix}0\\1\\-1\end{bmatrix}\Rightarrow(\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_1^{\mathrm{T}})\boldsymbol{C}=\begin{bmatrix}1&4&7\\1&1&1\\4&10&17\end{bmatrix}.
$$

### 3.2.3 高斯变换的舍入误差性质

如果 $\hat{\boldsymbol{\tau}}$ 是高斯向量 $\boldsymbol{\tau}$ 的计算值, 则容易证明

$$
\hat{\boldsymbol{\tau}}=\boldsymbol{\tau}+\boldsymbol{e},\quad |\boldsymbol{e}|\leqslant \mathbf{u}|\boldsymbol{\tau}|.
$$

设 $\hat{\boldsymbol{\tau}}$ 用于高斯变换修正且 $fl((\boldsymbol{I}-\hat{\boldsymbol{\tau}}\boldsymbol{e}_k^{\mathrm{T}})\boldsymbol{C})$ 是计算值, 则

$$
fl((\boldsymbol{I}-\hat{\boldsymbol{\tau}}\boldsymbol{e}_k^{\mathrm{T}})\boldsymbol{C})=(\boldsymbol{I}-\boldsymbol{\tau}\boldsymbol{e}_k^{\mathrm{T}})\boldsymbol{C}+\boldsymbol{E},
$$

其中

$$
|\boldsymbol{E}|\leqslant 3\mathbf{u}(|\boldsymbol{C}|+|\boldsymbol{\tau}||C(k,:)|)+O(\mathbf{u}^2).
$$

很明显, 如果 $\boldsymbol{\tau}$ 有大分量, 则修正的误差与 $|\boldsymbol{C}|$ 相比可能很大. 因此, 进行高斯变换时要小心, 这一点将在 3.4 节讨论.

### 3.2.4 上三角形化

设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$. 可找到高斯变换 $\boldsymbol{M}_1,\cdots,\boldsymbol{M}_{n-1}$ 使得 $\boldsymbol{M}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{A}=\boldsymbol{U}$ 是上三角形矩阵. 为说明这一点, 我们看一个 $n=3$ 的例子. 假设

$$
\boldsymbol{A}=\begin{bmatrix}1&4&7\\2&5&8\\3&6&10\end{bmatrix}.
$$

如果

$$
\boldsymbol{M}_1=\begin{bmatrix}1&0&0\\-2&1&0\\-3&0&1\end{bmatrix},
$$

则

$$
\boldsymbol{M}_1\boldsymbol{A}=\begin{bmatrix}1&4&7\\0&-3&-6\\0&-6&-11\end{bmatrix}.
$$

同样,

$$M_2=\begin{bmatrix}1&0&0\\0&1&0\\0&-2&1\end{bmatrix}\Rightarrow M_2(M_1A)=\begin{bmatrix}1&4&7\\0&-3&-6\\0&0&1\end{bmatrix}.$$

从此例可看出在第 $k$ 步

- 我们得到的矩阵 $A^{(k-1)}=M_{k-1}\cdots M_1A$ 在第 1 列至第 $k-1$ 列是上三角形的.
- $M_k$ 的乘子是基于 $A^{(k-1)}(k+1:n,k)$. 特别地, 我们需要 $a_{kk}^{(k-1)}\neq 0$.

注意到 $n-1$ 步之后就可完成上三角形化. 所以我们有

$$\begin{aligned}&k=1\\&\textbf{while }(A(k,k)\neq 0)\&(k\leqslant n-1)\\&\qquad \tau(k+1:n)=A(k+1:n,k)/A(k,k)\\&\qquad A(k+1:n,:)=A(k+1:n,:)-\tau(k+1:n)A(k,:)\\&\qquad k=k+1\\&\textbf{end}\end{aligned}\tag{3.2.2}$$

必须检查 $A(k,k)$ 以避免除零. 这些量 $(A(k,k))$ 称为主元, 它们的相对大小是至关重要的.

### 3.2.5 LU 分解

用矩阵语言, 如果 (3.2.2) 当 $k=n$ 时终止, 则它算出高斯变换 $M_1,\cdots,M_{n-1}$, 使得 $M_{n-1}\cdots M_1A=U$ 是上三角形矩阵. 容易验算, 如果 $M_k=I-\tau^{(k)}e_k^{\mathrm{T}}$, 则它的逆为 $M_k^{-1}=I+\tau^{(k)}e_k^{\mathrm{T}}$, 因而

$$A=LU,\tag{3.2.3}$$

其中

$$L=M_1^{-1}\cdots M_{n-1}^{-1}.\tag{3.2.4}$$

由于每个 $M_k^{-1}$ 都是单位下三角形矩阵, 故很明显 $L$ 是单位下三角形矩阵. 分解 (3.2.3) 称为 $A$ 的 LU 分解.

正如 (3.2.2) 中需要检查零主元所示, LU 分解可能不存在. 例如, 不可能找到 $l_{ij}$ 和 $u_{ij}$ 使得

$$\begin{bmatrix}1&2&3\\2&4&7\\3&5&3\end{bmatrix}=\begin{bmatrix}1&0&0\\l_{21}&1&0\\l_{31}&l_{32}&1\end{bmatrix}\begin{bmatrix}u_{11}&u_{12}&u_{13}\\0&u_{22}&u_{23}\\0&0&u_{33}\end{bmatrix}.$$

为说明这一点, 我们让对应元素相等, 发现 $u_{11}=1,u_{12}=2,l_{21}=2,u_{22}=0$ 和 $l_{31}=3$. 但是, 当比较两边的 (3,2) 元素时我们得到一个矛盾方程 $5=l_{31}u_{12}+l_{32}u_{22}=6$.

正如我们将证明的, (3.2.2) 中出现零主元与奇异主子矩阵是等价的.

**定理 3.2.1** 如果对 $k=1:n-1$, 行列式 $\det(A(1:k,1:k))\neq 0$, 则 $\boldsymbol{A}$ 有 LU 分解. 如果 $\boldsymbol{A}$ 非奇异且存在 LU 分解, 则 LU 分解是唯一的且 $\det(\boldsymbol{A})=u_{11}\cdots u_{nn}$.

**证明** 假定 (3.2.2) 的前 $k-1$ 步已完成. 在第 $k$ 步开始前 $\boldsymbol{A}$ 已被 $\boldsymbol{M}_{k-1}\cdots\boldsymbol{M}_1\boldsymbol{A}=\boldsymbol{A}^{(k-1)}$ 覆盖. 注意到 $a_{kk}^{(k-1)}$ 是第 $k$ 个主元. 由于高斯变换是单位下三角形矩阵, 从该等式的主 $k\times k$ 部分可看出, $\det(A(1:k,1:k))=a_{11}^{(k-1)}\cdots a_{kk}^{(k-1)}$. 所以, 如果 $A(1:k,1:k)$ 非奇异, 则第 $k$ 个主元非零.

至于唯一性, 如果 $\boldsymbol{A}=\boldsymbol{L}_1\boldsymbol{U}_1$ 和 $\boldsymbol{A}=\boldsymbol{L}_2\boldsymbol{U}_2$ 是非奇异矩阵 $\boldsymbol{A}$ 的两个 LU 分解, 则 $\boldsymbol{L}_2^{-1}\boldsymbol{L}_1=\boldsymbol{U}_2\boldsymbol{U}_1^{-1}$. 由于 $\boldsymbol{L}_2^{-1}\boldsymbol{L}_1$ 是单位下三角形矩阵而 $\boldsymbol{U}_2\boldsymbol{U}_1^{-1}$ 是上三角形矩阵, 故知这两个矩阵都必须是单位矩阵. 因此 $\boldsymbol{L}_1=\boldsymbol{L}_2,\boldsymbol{U}_1=\boldsymbol{U}_2$.

最后, 如果 $\boldsymbol{A}=\boldsymbol{L}\boldsymbol{U}$, 则 $\det(\boldsymbol{A})=\det(\boldsymbol{L}\boldsymbol{U})=\det(\boldsymbol{L})\det(\boldsymbol{U})=\det(\boldsymbol{U})=u_{11}\cdots u_{nn}$. □

### 3.2.6 一些实际细节

从实际应用的角度, 能对 (3.2.2) 有几种改进. 首先, 由于从第 1 列至第 $k-1$ 列已化为上三角形, 高斯变换只需作用于从第 $k$ 列至第 $n$ 列. 当然, 我们不需将第 $k$ 个高斯变换作用于 $A(:,k)$, 因为其结果是知道的. 另一个值得注意的事项是, 对应于 $\boldsymbol{M}_k$ 的乘子可储存在已消为零的位置, 即 $A(k+1:n,k)$. 利用这些改动, 我们得到 (3.2.2) 的如下形式.

**算法 3.2.1 (外积高斯消去法)** 设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 满足 $A(1:k,1:k)$ 对 $k=1:n-1$ 非奇异. 本算法计算分解 $\boldsymbol{M}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{A}=\boldsymbol{U}$, 其中 $\boldsymbol{U}$ 是上三角形矩阵, 每个 $\boldsymbol{M}_k$ 都是高斯变换. $\boldsymbol{U}$ 储存于 $\boldsymbol{A}$ 的上三角部分. 对应于 $\boldsymbol{M}_k$ 的乘子储存于 $A(k+1:n,k)$, 即 $A(k+1:n,k)=-M_k(k+1:n,k)$.

```
for k = 1 : n − 1
    rows=k + 1 : n
    A(rows, k) = A(rows, k)/A(k, k)
    A(rows, rows) = A(rows, rows) − A(rows, k)A(k, rows)
end
```

此算法需要 $\dfrac{2}{3}n^3$ 个 flop, 是高斯消去法的若干种形式之一. 注意到在 $k$ 循环的每一步中执行的是一外积运算.

### 3.2.7 $L$ 存于何处?

算法 3.2.1 以乘子的方式表示 $\boldsymbol{L}$. 确切地说, 设 $\boldsymbol{\tau}^{(k)}$ 是对应于 $\boldsymbol{M}_k$ 的乘子, 当算法终止时 $A(k+1:n,k)=\boldsymbol{\tau}^{(k)}$. 矩阵计算中的可喜的 "意外" 是, 如果 $\boldsymbol{L}=\boldsymbol{M}_1^{-1}\cdots\boldsymbol{M}_{n-1}^{-1}$, 则 $L(k+1:n,k)=\boldsymbol{\tau}^{(k)}$. 这从仔细考查 $\boldsymbol{L}$ 的乘积定义就可得到. 事实上

$$\boldsymbol{L}=(\boldsymbol{I}+\boldsymbol{\tau}^{(1)}\boldsymbol{e}_1^{\mathrm{T}})\cdots(\boldsymbol{I}+\boldsymbol{\tau}^{(n-1)}\boldsymbol{e}_{n-1}^{\mathrm{T}})=I+\sum_{k=1}^{n-1}\boldsymbol{\tau}^{(k)}\boldsymbol{e}_k^{\mathrm{T}}.$$

由于 $A(k+1:n,k)$ 储存乘子 $\tau^{(k)}$ 的第 $k$ 个向量, 故知对一切 $i>k$, $A(i,k)$ 存储 $l_{ik}$.

### 3.2.8 解线性方程组

一旦 $\boldsymbol{A}$ 已用算法 3.2.1 被分解, 则 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 存于数组 $\boldsymbol{A}$. 然后, 我们可用 3.1 节的方法解三角方程组 $\boldsymbol{Ly}=\boldsymbol{b}$ 和 $\boldsymbol{Ux}=\boldsymbol{y}$, 得到 $\boldsymbol{Ax}=\boldsymbol{b}$ 之解.

**例 3.2.2** *如果算法 3.2.1 用于*

$$\boldsymbol{A}=\begin{bmatrix}1&4&7\\2&5&8\\3&6&10\end{bmatrix}=\begin{bmatrix}1&0&0\\2&1&0\\3&2&1\end{bmatrix}\begin{bmatrix}1&4&7\\0&-3&-6\\0&0&1\end{bmatrix},$$

*则完成后可得*

$$\boldsymbol{A}=\begin{bmatrix}1&4&7\\2&-3&-6\\3&2&1\end{bmatrix}.$$

*如果 $\boldsymbol{b}=(1,1,1)^{\mathrm{T}}$, 则 $\boldsymbol{y}=(1,-1,0)^{\mathrm{T}}$ 是 $\boldsymbol{Ly}=\boldsymbol{b}$ 之解. 再解 $\boldsymbol{Ux}=\boldsymbol{y}$ 得 $\boldsymbol{x}=\left(-\frac{1}{3},\frac{1}{3},0\right)^{\mathrm{T}}$.*

### 3.2.9 其他形式

与矩阵乘法一样, 高斯消去法也是一个三重求和, 可以有不同顺序. 如果算法 3.2.1 的外积计算是逐行计算, 则它对应于高斯消去法的 $kij$ 形式:

```
for k = 1 : n − 1
    A(k + 1 : n, k) = A(k + 1 : n, k)/A(k, k)
    for i = k + 1 : n
        for j = k + 1 : n
            A(i, j) = A(i, j) − A(i, k)A(k, j)
        end
    end
end
```

其他的五种形式为 $kji, ikj, ijk, jik$ 和 $jki$. 这些实现方式的后三种之特点是一系列的 gaxpy 和向前消去. 在此方式下, 高斯变换不像外积形式那样马上作用于 $\boldsymbol{A}$. 事实上, 这些变换延后了. 在第 $j$ 步之前, 初始的 $A(:,j)$ 根本就没有动. 在算法第 $j$ 步, $A(:,j)$ 被 $\boldsymbol{M}_{j-1}\cdots\boldsymbol{M}_1A(:,j)$ 覆盖, 第 $j$ 个高斯变换也求出来了.

更精确地说, 设 $1\leqslant j\leqslant n-1$, 假定 $L(:,1:j-1)$ 和 $U(1:j-1,1:j-1)$ 已知, 这意味着 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 的前 $j-1$ 列已求出来了. 为得到 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 的第 $j$ 列, 我们令方程 $\boldsymbol{A}=\boldsymbol{LU}$ 的第 $j$ 列相等: $A(:,j)=LU(:,j)$. 由此, 我们有

$$A(1:j-1,j)=L(1:j-1,1:j-1)U(1:j-1,j),$$

$$A(j:n,j)=\sum_{k=1}^{j}L(j:n,k)U(k,j).$$

第一个方程是下三角方程组, 可从它解出 $U(1:j-1,j)$. 完成这点之后, 可改写第二个方程来计算 $U(j,j)$ 和 $L(j+1:n,j)$. 事实上, 我们令

$$\begin{aligned}v(j:n)&=A(j:n,j)-\sum_{k=1}^{j-1}L(j:n,k)U(k,j)\\&=A(j:n,j)-L(j:n,1:j-1)U(1:j-1,j),\end{aligned}$$

则 $L(j+1:n,j)=v(j+1:n)/v(j)$, 以及 $U(j,j)=v(j)$. 于是计算 $L(j+1:n,j)$ 是一个加权 gaxpy, 我们有

```
L = I; U = 0
for j = 1 : n
    if j = 1
        v(j : n) = A(j : n, j)
    else
        从 L(1 : j − 1, 1 : j − 1)z = A(1 : j − 1, j) 解出 z
            令 U(1 : j − 1, j) = z                                    (3.2.5)
        v(j : n) = A(j : n, j) − L(j : n, 1 : j − 1)z
    end
    if j < n
        L(j + 1 : n, j) = v(j + 1 : n)/v(j)
    end
    U(j, j) = v(j)
end
```

高斯消去法的此形式主要是向前消去和 gaxpy, 它与算法 3.2.1 一样需要 $\frac{2}{3}n^3$ 个 flop.

### 3.2.10 分块 LU

高斯消去法可组织得让矩阵乘法是主要的运算. 这一分块方法之导入的关键是把 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 划分为如下形式

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_{11} & \boldsymbol{A}_{12}\\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22}\end{bmatrix}\begin{matrix}r\\ n-r\end{matrix}$$
$$\qquad\quad r\qquad n-r$$

其中 $r$ 是分块参数. 假设我们已算出 LU 分解 $\boldsymbol{A}_{11}=\boldsymbol{L}_{11}\boldsymbol{U}_{11}$, 则可解多右端项三角方程组 $\boldsymbol{L}_{11}\boldsymbol{U}_{12}=\boldsymbol{A}_{12}$ 和 $\boldsymbol{L}_{21}\boldsymbol{U}_{11}=\boldsymbol{A}_{21}$ 分别得到 $\boldsymbol{U}_{12}$ 和 $\boldsymbol{L}_{21}$. 从而有

$$\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{I}_{n-r} \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{0} & \widetilde{\boldsymbol{A}} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{11} & \boldsymbol{U}_{12} \\ \boldsymbol{0} & \boldsymbol{I}_{n-r} \end{bmatrix},$$

其中 $\widetilde{\boldsymbol{A}} = \boldsymbol{A}_{22} - \boldsymbol{L}_{21}\boldsymbol{U}_{12}$. 矩阵 $\widetilde{\boldsymbol{A}}$ 是 $\boldsymbol{A}_{11}$ 关于 $\boldsymbol{A}$ 的 Schur 补. 注意到, 如果 $\widetilde{\boldsymbol{A}} = \boldsymbol{L}_{22}\boldsymbol{U}_{22}$ 是 $\widetilde{\boldsymbol{A}}$ 之 LU 分解, 则

$$\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{0} & \tilde{\boldsymbol{A}} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{11} & \boldsymbol{U}_{12} \\ \boldsymbol{0} & \boldsymbol{U}_{22} \end{bmatrix}$$

就是 $\boldsymbol{A}$ 的 LU 分解. 于是当 $\boldsymbol{L}_{11}, \boldsymbol{L}_{21}, \boldsymbol{U}_{11}$ 和 $\boldsymbol{U}_{12}$ 计算出之后, 我们继续对 (2,2) 块 $\widetilde{\boldsymbol{A}}$ 进行 3 级修正.

**算法 3.2.2 (分块外积 LU)**　设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 且 $\det(A(1:k,1:k))$ 对 $k=1:n-1$ 非零. 设 $r$ 满足 $1 \leqslant r \leqslant n$. 本算法用秩 $r$ 修正计算 $\boldsymbol{A} = \boldsymbol{LU}$. 算法完成后, $A(i,j)$ 当 $i > j$ 时被 $L(i,j)$ 覆盖, 当 $j \geqslant i$ 时被 $U(i,j)$ 覆盖.

$\lambda = 1$
**while** $\lambda \leqslant n$
　　$\mu = \min(n, \lambda + r - 1)$
　　用算法 3.2.1 覆盖 $A(\lambda:\mu, \lambda:\mu)$, 其 LU 分解因子为 $\widetilde{\boldsymbol{L}}$ 和 $\widetilde{\boldsymbol{U}}$
　　解 $\widetilde{\boldsymbol{L}}\boldsymbol{Z} = A(\lambda:\mu, \mu+1:n)$ 得到 $\boldsymbol{Z}$ 且用 $\boldsymbol{Z}$ 覆盖 $A(\lambda:\mu, \mu+1:n)$
　　解 $\boldsymbol{W}\widetilde{\boldsymbol{U}} = A(\mu+1:n, \lambda:\mu)$ 得到 $\boldsymbol{W}$ 且用 $\boldsymbol{W}$ 覆盖 $A(\mu+1:n, \lambda:\mu)$
　　$A(\mu+1:n, \mu+1:n) = A(\mu+1:n, \mu+1:n) - \boldsymbol{WZ}$
　　$\lambda = \mu + 1$
**end**

此算法需要 $\frac{2}{3}n^3$ 个 flop.

与 3.1.5 节的讨论类似, 我们讨论此方法的 3 级比例. 假定 $r$ 足够大, 所用的计算机能以 “3 级速度” 计算矩阵的相乘修正 $A(\mu+1:n, \mu+1:n) = A(\mu+1:n, \mu+1:n) - WZ$. 为明确起见, 假定 $n = rN$. 非 3 级的浮点运算仅仅是 $r \times r$ 的 LU 分解 $A(\lambda:\mu, \lambda:\mu) = \widetilde{L}\widetilde{U}$. 在整个计算中. 一共有 $N$ 个这样的分解, 故知 3 级比例为

$$1 - \frac{N(2r^3/3)}{2n^3/3} = 1 - \frac{1}{N^2}.$$

这样, 当 $N$ 很大时几乎所有的运算都是矩阵相乘. 正如我们已提到, 这样就保证了在许多计算环境下算法都是高性能的.

### 3.2.11　长方矩阵的 LU 分解

对长方矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 也可进行 LU 分解. $m > n$ 时的例子可如下给出:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 3 & 1 \\ 5 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & -2 \end{bmatrix}.$$

而

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \end{bmatrix}$$

说明了 $m < n$ 的情形. 如果 $A(1:k, 1:k)$ 对 $k = 1:\min(m,n)$ 都非奇异的, 则 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 就一定存在 LU 分解.

上面关于方阵的 LU 分解算法稍加改动就可用于长方矩阵. 例如, 对 $m > n$, 算法 3.2.1 可修改为

```
for k = 1 : n
    rows=k + 1 : m
    A(rows,k)=A(rows,k)/A(k, k)
    if k < n
        cols=k + 1 : n
        A(rows,cols)=A(rows,cols)−A(rows,k)A(k, cols)
    end
end
```

此算法需要 $mn^2 - n^3/3$ 个 flop.

### 3.2.12 一点不足

正如我们所知, 高斯消去法在前 $n-1$ 个主子矩阵奇异时不能进行. 这种情况对有些非常简单的矩阵发生, 如

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

$\boldsymbol{A}$ 的 2 范数条件非常好, 但它的主子矩阵奇异, 故不存在 LU 分解.

很明显, 要使高斯消去法有效地用于求解一般线性方程组, 我们就要对它进行改进. 下一节给出的误差分析指明哪些改进是必需的.

## 习 题

**3.2.1** 设 $\boldsymbol{A}(\varepsilon) \in \mathbb{R}^{n\times n}$ 的元素是标量 $\varepsilon$ 的连续可微函数, 假设 $\boldsymbol{A} \equiv \boldsymbol{A}(0)$ 以及它的所有主子矩阵非奇异. 证明: 对充分小的 $\varepsilon$, 矩阵 $\boldsymbol{A}(\varepsilon)$ 有 LU 分解 $\boldsymbol{A}(\varepsilon) = \boldsymbol{L}(\varepsilon)\boldsymbol{U}(\varepsilon)$ 且 $\boldsymbol{L}(\varepsilon)$ 和 $\boldsymbol{U}(\varepsilon)$ 都是连续可微的.

**3.2.2** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 的划分为

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix},$$

其中 $A_{11}$ 是 $r\times r$ 矩阵. 假定 $\boldsymbol{A}_{11}$ 非奇异. 矩阵 $\boldsymbol{S} = \boldsymbol{A}_{22} - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}$ 是在 $\boldsymbol{A}$ 中 $\boldsymbol{A}_{11}$ 的 Schur 补. 证明: 如果 $\boldsymbol{A}_{11}$ 有 LU 分解, 则算法 3.2.1 执行到第 $r$ 步之后, $A(r+1:n, r+1:n)$ 储存了 $\boldsymbol{S}$. 为什么在 (3.2.5) 的 $r$ 步之后可得到 $S$?

**3.2.3** 设 $A \in \mathbb{R}^{n\times n}$ 有 LU 分解. 说明为什么通过计算 $n \times (n+1)$ 矩阵 $[A \quad b]$ 的 LU 分解而不储存乘子就可求解 $Ax = b$.

**3.2.4** 叙述一个变形的高斯消去法. 它按 $n:-1:2$ 的顺序将 $A$ 列消去. 从而得到分解 $A = UL$, 其中 $U$ 是单位上三角形矩阵, $L$ 是下三角形矩阵.

**3.2.5** $\mathbb{R}^{n\times n}$ 中形如 $N(y,k) = I - ye_k^{\mathrm{T}}$ $(y \in \mathbb{R}^n)$ 的矩阵称为 Gauss-Jordan 变换. (a) 假定 $N(y,k)^{-1}$ 存在, 给出其公式. (b) 给定 $x \in \mathbb{R}^n$, 什么条件下存在 $y$ 使得 $N(y,k)x = e_k$? (c) 给出一个应用 Gauss-Jordan 变换的将 $A$ 用 $A^{-1}$ 覆盖的算法. $A$ 满足什么条件时能保证方法成功?

**3.2.6** 将 (3.2.5) 推广到 $A$ 的行数大于列数的情形.

**3.2.7** 说明 (3.2.5) 中 $A$ 可被 $L$ 和 $U$ 覆盖. 重新组织三重循环使得对数据的读取是整体间.

**3.2.8** 给出高斯消去法的一种形式, 其三重循环之最内层是点积计算.

## 本节注释与参考文献

Schur 补 (习题 3.2.2) 广泛见于各种应用. 关于它的理论与应用, 可见:

R. W. Cottle(1974). "Manifestations of the Schur Complement," *Lin. Alg. and Its Applic.* 8, 189–211.

Schur 补在一些应用领域里也称 "高斯变换". Gauss-Jordan 变换 (习题 3.2.5) 的用处之详细讨论可见 Fox(1964), 也可参阅:

T. Dekker and W. Hoffman(1989). "Rehabilitation of the Gauss-Jordan Algorithm," *Numer. Math.* 54, 591–599.

正如我们所指出的, 内积形式的高斯消去法已被人所知和所用很长时间了. Crout 和 Doolittle 的名字是与这些 $ijk$ 技巧有关联的. 它们在桌面计算器时代是常用的, 因为这比高斯消去法少很多中间计算结果. 这些方法今天仍有吸引力是因为它们在实现上可利用累加内积. 这方面的讨论可见 Fox(1964) 和 Stewart(1973, 131–139 页). 也可见:

G. E. Forsythe(1960). "Crout with Pivoting," *Comm. ACM* 3, 507–508.

W. M. McKeeman(1962). "Crout with Equilibration and Iteration," *Comm. ACM.* 5, 553–555.

LU 分解的循环顺序以及分块技巧等的讨论可见:

J. J. Dongarra, F. G. Gustavson, and A. Karp(1984). "Implementing Linear Algebra Algorithms for Dense Matrices on a Vector Pipeline Machine," *SIAM Review* 26, 91–112.

J. M .Ortega(1988). "The $ijk$ Forms of Factorization Methods I: Vector Computers," *Parallel Computers* 7, 135–147.

D. H. Bailey, K. Lee, and H. D. Simon (1991). "Using Strassen's Algorithm to Accelerate the Solution of Linear Systems," *J. Supercomputing* 4, 357–371.

J. W. Demmel, N. J. Higham, and R. S. Schreiber (1995). "Stability of Block LU Factorization," *Numer. Lin. Alg. with Applic.* 2, 173–190.

## 3.3 高斯消去法的舍入误差分析

我们现在讨论上两节的算法用于求解线性方程组 $Ax=b$ 时舍入误差的影响. 关于高斯消去法舍入误差的更详细的讨论由 Higham(1996) 给出.

在开始分析前, 一件有用的事是讨论几乎理想的情形, 即除了储存 $A$ 和 $b$ 之外整个计算过程都没有舍入误差. 这样, 当 $fl(b)=b+e$ 且 $fl(A)=A+E$ 非奇异时, 我们假定 $\hat{x}$ 满足

$$(A+E)\hat{x}=(b+e),\quad \|\mathbf{E}\|_\infty \leqslant \mathbf{u}\|A\|_\infty,\quad \|e\|_\infty \leqslant \mathbf{u}\|b\|_\infty. \tag{3.3.1}$$

也就是说, $\hat{x}$ 是一个 "附近" 问题的精确解. 而且, 如果 $\mathbf{u}\kappa_\infty(A)\leqslant \frac{1}{2}$, 则利用定理 2.7.2 可证明

$$\frac{\|x-\hat{x}\|_\infty}{\|x\|_\infty}\leqslant 4\mathbf{u}\kappa_\infty(A). \tag{3.3.2}$$

界 (3.3.1) 和界 (3.3.2) 是 "最好" 的范数界. 任何对需要储存 $A$ 和 $b$ 的线性方程组求解方法的一般性 $\infty$ 范数误差分析不可能得到更好的界. 因此, 如果 $A$ 的病态已相当于机器精度, 即 $\mathbf{u}\kappa_\infty(A)\approx 1$, 则我们不应抱怨算法给出的 $\hat{x}$ 不精确.

### 3.3.1 LU 分解的误差

下面讨论高斯消去法的误差界与上述理想界的关系. 为方便起见, 我们考虑 $\infty$ 范数且讨论算法 3.2.3, 即外积形式. 我们所导出的误差界也适用于 gaxpy 形式的算法 3.2.4.

我们第一个任务是量化计算三角分解时的舍入误差.

**定理 3.3.1** *假定 $A$ 是一个 $n\times n$ 浮点矩阵. 如果算法 3.2.3 中不出现零主元情况, 则计算出来的三角形矩阵 $\hat{L}$ 和 $\hat{U}$ 满足*

$$\hat{L}\hat{U}=A+H, \tag{3.3.3}$$

$$|H|\leqslant 3(n-1)\mathbf{u}(|A|+|\hat{L}||\hat{U}|)+O(\mathbf{u}^2). \tag{3.3.4}$$

**证明** 对 $n$ 进行归纳. 定理对 $n=1$ 显然成立. 假定它对 $(n-1)\times(n-1)$ 浮点矩阵是成立的. 设

$$A=\begin{bmatrix}\alpha & w^{\mathrm{T}}\\ v & B\end{bmatrix}\begin{matrix}1\\ n-1\end{matrix}$$
$$\quad\ \ 1\qquad n-1$$

则算法的第一步产生 $\hat{z}=fl(v/a)$ 和 $\hat{A}_1=fl(B-\hat{z}w^{\mathrm{T}})$. 于是我们有

$$\hat{z}=\frac{1}{\alpha}v+f,\quad |f|\leqslant \mathbf{u}\frac{|v|}{|\alpha|}, \tag{3.3.5}$$

$$\hat{A}_1=B-\hat{z}w^{\mathrm{T}}+F,\quad |F|\leqslant 2\mathbf{u}(|B|+|\hat{z}||w|^{\mathrm{T}})+O(\mathbf{u}^2). \tag{3.3.6}$$

算法现在需要计算 $\hat{A}_1$ 的 LU 分解. 由归纳法, 计算出来的 $\hat{A}_1$ 的近似分解 $\hat{L}_1$ 和 $\hat{U}_1$ 满足

$$\hat{L}_1\hat{U}_1 = \hat{A}_1 + H_1, \tag{3.3.7}$$

$$|H_1| \leqslant 3(n-2)\mathbf{u}(|\hat{A}_1| + |\hat{L}_1||\hat{U}_1|) + O(\mathbf{u}^2). \tag{3.3.8}$$

于是

$$\begin{aligned}\hat{L}\hat{U} &\equiv \begin{bmatrix} 1 & 0 \\ \hat{z} & \hat{L}_1 \end{bmatrix}\begin{bmatrix} \alpha & w^{\mathrm{T}} \\ 0 & \hat{U}_1 \end{bmatrix} \\ &= A + \begin{bmatrix} 0 & 0 \\ \alpha f & H_1 + F \end{bmatrix} \equiv A + H.\end{aligned}$$

从 (3.3.6) 知

$$|\hat{A}_1| \leqslant (1+2\mathbf{u})(|B| + |\hat{z}||w|^{\mathrm{T}}) + O(\mathbf{u}^2),$$

于是从 (3.3.7) 和 (3.3.8) 我们有

$$|H_1 + F| \leqslant 3(n-1)\mathbf{u}(|B| + |\hat{z}||w|^{\mathrm{T}} + |\hat{L}_1||\hat{U}_1|) + O(\mathbf{u}^2).$$

由于 $|\alpha f| \leqslant \mathbf{u}|v|$, 容易验证

$$|H| \leqslant 3(n-1)\mathbf{u}\left\{\begin{bmatrix} |\alpha| & |w|^{\mathrm{T}} \\ |v| & |B| \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ |\hat{z}| & |\hat{L}_1| \end{bmatrix}\begin{bmatrix} |\alpha| & |w|^{\mathrm{T}} \\ 0 & |\hat{U}_1| \end{bmatrix}\right\} + O(\mathbf{u}^2).$$

故知定理成立. □

我们指出, 如果 $A$ 是 $m \times n$ 矩阵, 则定理把 (3.3.4) 中的 $n$ 用 $\min\{n, m\}$ 替代后仍成立.

### 3.3.2 非精确三角方程组的求解

接下来我们考查当 $\hat{L}$ 和 $\hat{U}$ 用于 3.1 节中三角方程组求解时舍入误差的影响.

**定理 3.3.2** 设 $\hat{L}$ 和 $\hat{U}$ 是 $n \times n$ 浮点矩阵 $A$ 通过算法 3.2.3 或算法 3.2.4 所求得的 LU 因子. 假设 3.1 节中的方法用来计算 $\hat{L}y = b$ 和 $\hat{U}x = \hat{y}$ 之解 $\hat{y}$ 和 $\hat{x}$. 则 $(A+E)\hat{x} = b$, 其中

$$|E| \leqslant n\mathbf{u}(3|A| + 5|\hat{L}||\hat{U}|) + O(\mathbf{u}^2). \tag{3.3.9}$$

**证明** 从 (3.1.1) 和 (3.1.2) 我们有

$$(\hat{L} + F)\hat{y} = b, \quad |F| \leqslant n\mathbf{u}|\hat{L}| + O(\mathbf{u}^2),$$

$$(\hat{U} + G)\hat{x} = \hat{y}, \quad |G| \leqslant n\mathbf{u}|\hat{U}| + O(\mathbf{u}^2).$$

于是

$$(\hat{L} + F)(\hat{U} + G)\hat{x} = (\hat{L}\hat{U} + F\hat{U} + \hat{L}G + FG)\hat{x} = b.$$

从定理 3.3.1, 知

$$\hat{L}\hat{U} = A + H,$$

且 $|\boldsymbol{H}| \leqslant 3(n-1)\mathbf{u}(|\boldsymbol{A}|+|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|)+O(\mathbf{u}^2)$. 所以, 通过定义

$$\boldsymbol{E}=\boldsymbol{H}+\boldsymbol{F}\hat{\boldsymbol{U}}+\hat{\boldsymbol{L}}\boldsymbol{G}+\boldsymbol{F}\boldsymbol{G}$$

则有 $(\boldsymbol{A}+\boldsymbol{E})\hat{x}=b$. 而且

$$\begin{aligned}|\boldsymbol{E}| &\leqslant |\boldsymbol{H}|+|\boldsymbol{F}||\hat{\boldsymbol{U}}|+|\hat{\boldsymbol{L}}||\boldsymbol{G}|+O(\mathbf{u}^2)\\ &\leqslant 3n\mathbf{u}(|\boldsymbol{A}|+|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|)+2n\mathbf{u}(|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|)+O(\mathbf{u}^2).\end{aligned}$$

□

要是 $|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|$ 这一项不太大的话, (3.3.9) 与 (3.3.1) 的理想界相差不大. (因子 $n$ 无关紧要, 见 2.4.6 节中 Wilkinson 的摘文.) 但有可能 $|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|$ 很大, 因为高斯消去法并不能排除小主元的可能性. 如果碰到很小的主元, 则 $\hat{\boldsymbol{L}}$ 和 $\hat{\boldsymbol{U}}$ 可能有很大的数.

我们强调一下, 小主元并不一定是由于病态所致. 如 $\boldsymbol{A}=\begin{bmatrix}\varepsilon & 1\\ 1 & 0\end{bmatrix}$ 所示. 所以, 甚至对于良态问题高斯消去法也可能给出任意差的结果. 此方法是不稳定的.

为了克服算法的这一缺点, 有必要在消去的过程中交换行和交换列, 使得计算中得到的数适当地有界. 这一思想将在下一节中讨论.

**例 3.3.1** 设 $\beta=10, t=3$. 浮点运算用于求解

$$\begin{bmatrix}0.001 & 1.00\\ 1.00 & 2.00\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\begin{bmatrix}1.00\\ 3.00\end{bmatrix}.$$

用高斯消去法, 我们有

$$\hat{\boldsymbol{L}}=\begin{bmatrix}1 & 0\\ 1000 & 1\end{bmatrix},\quad \hat{\boldsymbol{U}}=\begin{bmatrix}0.001 & 1\\ 0 & -1000\end{bmatrix},$$

$$\hat{\boldsymbol{L}}\hat{\boldsymbol{U}}=\begin{bmatrix}0.001 & 1\\ 1 & 2\end{bmatrix}+\begin{bmatrix}0 & 0\\ 0 & -2\end{bmatrix}\equiv \boldsymbol{A}+\boldsymbol{H}.$$

而且 $6\begin{bmatrix}10^{-6} & 0.001\\ 10^{-3} & 1.0001\end{bmatrix}$ 是 (3.3.4) 中的界矩阵, 它没有过高估计 $|\boldsymbol{H}|$. 如果我们用 3.1 节的三角方程组求解方法继续解决此问题, 而且假定精度同上, 则可得到计算结果 $\hat{\boldsymbol{x}}=(0,1)^{\mathrm{T}}$. 这与精确解 $\boldsymbol{x}=(1.002\cdots,0.998\cdots)^{\mathrm{T}}$ 差别很大.

## 习 题

**3.3.1** 证明如果在定理 3.3.1 中把 $\boldsymbol{A}$ 是浮点矩阵的假设去掉, 则 (3.3.4) 将在把系数 "3" 换为 "4" 之后成立.

**3.3.2** 设 $\boldsymbol{A}$ 是 $n\times n$ 矩阵且 $\hat{\boldsymbol{L}}$ 和 $\hat{\boldsymbol{U}}$ 由算法 3.2.1 产生. (a) 计算 $\||\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|\|_\infty$ 需要多少 flop? (b) 证明 $fl(|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|)\leqslant(1+2n\mathbf{u})|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|+O(\mathbf{u}^2)$.

**3.3.3** 设 $\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$. 证明: 如果 $\boldsymbol{e}=\boldsymbol{x}-\hat{\boldsymbol{x}}$ (误差), $\boldsymbol{r}=\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}$ (余量), 则

$$\frac{\|\boldsymbol{r}\|}{\|\boldsymbol{A}\|}\leqslant\|\boldsymbol{e}\|\leqslant\|\boldsymbol{A}^{-1}\|\|\boldsymbol{r}\|.$$

假定矩阵范数与向量范数是一致的.

**3.3.4** 用十进制的两位浮点运算, 计算

$$A = \begin{bmatrix} 7 & 6 \\ 9 & 8 \end{bmatrix}$$

的 LU 分解. 对于这个例子, (3.3.3) 中的 $H$ 是什么?

### 本节注释与参考文献

关于高斯消去法的最原始的舍入误差分析出现于:

J. H. Wilkinson(1961). "Error Analysis of Direct Methods of Matrix Inversion," *J. ACM* 8, 281–330.

长期以来, 关于误差分析的界之各种改进以及分析的简化有大量工作, 见:

B. A. Chartres and J. C. Geuder (1967). "Computable Error Bounds for Direct Solution of Linear Equations," *J. ACM* 14, 63–71.

J. K. Reid (1971). "A Note on the Stability of Gaussian Elimination," *J. Inst. Math. Applic.* 8, 374–375.

C. C. Paige (1973). "An Error Analysis of a Method for Solving Matrix Equations," *Math. Comp.* 27, 355–359.

C. de Boor and A. Pinkus (1977). "A Backward Error Analysis for Totally Positive Linear Systems," *Numer. Math.* 27, 485–490.

H. H. Robertson (1977). "The Accuracy of Error Estimates for Systems of Linear Algebraic Equations," *J. Inst. Math. Applic.* 20, 409–414.

J. J. Du Croz and N. J. Higham (1992). "Stability of Methods for Matrix Inversion," *IMA J. Num. Anal.* 12, 1–19.

## 3.4 选主元法

上一节的分析表明, 我们应采取措施使计算的三角形因子 $\hat{L}$ 和 $\hat{U}$ 没有太大的元素. 例子

$$A = \begin{bmatrix} 0.0001 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10\,000 & 1 \end{bmatrix} \begin{bmatrix} 0.0001 & 1 \\ 0 & -9999 \end{bmatrix} = LU$$

准确地给出了困难的根源, 即相对很小的主元. 克服此困难的方法之一是交换行. 在我们的例子中, 如果 $A$ 是置换矩阵

$$P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

则

$$PA = \begin{bmatrix} 1 & 1 \\ 0.0001 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0.0001 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 0.9999 \end{bmatrix} = LU.$$

此时, 三角形因子之元素都是可接受的小元素.

在此节, 我们阐明如何决定 $\boldsymbol{A}$ 的置换形式使其有一基本稳定的 LU 分解. 这有几种做法, 每一种对应于不同的选主元技巧. 我们主要讨论部分选主与全选主. 将讨论这些技巧的有效实现以及它们的性质. 首先我们讨论置换矩阵乘法.

### 3.4.1 置换矩阵

本节要讨论的高斯消去法的稳定化涉及如交换矩阵的两行这样的数据流动. 为使所有的计算都用 "矩阵语言" 来描述, 有必要熟悉置换矩阵. 一个**置换矩阵**就是把单位矩阵的行重新排列后所得的矩阵, 例如

$$\boldsymbol{P} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

一个 $n \times n$ 置换矩阵永远不必显式储存, 用一个 $n$ 维整数向量 $\boldsymbol{p}$ 表示一个一般置换矩阵 $\boldsymbol{P}$ 要有效得多. 一种方法是令 $p(k)$ 是 $\boldsymbol{P}$ 的第 $k$ 行中唯一个 "1" 的列指标. 于是, $p = [4\ \ 1\ \ 3\ \ 2]$ 就是上面 $\boldsymbol{P}$ 的一个很好的编码. 也可以按 $\boldsymbol{P}$ 每列中 "1" 的位置来编码. 在这种方式下对上面 $\boldsymbol{P}$ 有 $p = [2\ \ 4\ \ 3\ \ 1]$.

设 $\boldsymbol{P}$ 是一置换矩阵. $\boldsymbol{A}$ 是一矩阵, 则 $\boldsymbol{PA}$ 是 $\boldsymbol{A}$ 的行的置换, $\boldsymbol{AP}$ 是 $\boldsymbol{A}$ 的列的置换. 置换矩阵是正交的, 所以, 如果 $P$ 是置换矩阵, 则 $\boldsymbol{P}^{-1} = \boldsymbol{P}^{\mathrm{T}}$. 置换矩阵的乘积是置换矩阵.

本节我们特别关注交换置换矩阵, 它们是将单位矩阵中两行互相对换而得到的. 例如

$$\boldsymbol{E} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

交换置换矩阵可用来描述行和列的互换. 对于上面的 $4 \times 4$ 例子, $\boldsymbol{EA}$ 是互换 $\boldsymbol{A}$ 的第 1 行和 4 行. 同样, $\boldsymbol{AE}$ 是互换 $\boldsymbol{A}$ 的第 1 列和第 4 列.

如果 $\boldsymbol{P} = \boldsymbol{E}_n \cdots \boldsymbol{E}_1$ 且每个 $\boldsymbol{E}_k$ 是将单位矩阵的第 $k$ 行与 $p(k)$ 行互换, 则 $p(1:n)$ 是 $\boldsymbol{P}$ 的一个有用的向量编码. 事实上, 对 $\boldsymbol{x} \in \mathbb{R}^n$, 它被 $\boldsymbol{Px}$ 覆盖:

**for** $k = 1:n$

  $x(k) \leftrightarrow x(p(k))$

**end**

这里 "$\leftrightarrow$" 记号表示对调, 由于每个 $E_k$ 都对称且 $\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{E}_1 \cdots \boldsymbol{E}_n$, 这个表达式可用来将 $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{x}$ 覆盖 $\boldsymbol{x}$:

**for** $k = n:-1:1$

$x(k) \leftrightarrow x(p(k))$

**end**

应该指出的是, 在置换运算中没有浮点运算. 但是, 置换矩阵算子常常涉及数据的非常规移动, 可能带来相当大的计算负担.

### 3.4.2　列选主元: 基本思想

我们证明在 LU 计算中用交换置换可保证所有乘子之绝对值不大于 1. 假定

$$\boldsymbol{A}=\begin{bmatrix}3 & 17 & 10\\ 2 & 4 & -2\\ 6 & 18 & -12\end{bmatrix}.$$

为了使第一个高斯变换乘子尽可能小, 我们用行互换使 $a_{11}$ 是第 1 列最大的元素. 所以, 如果 $\boldsymbol{E}_1$ 是交换置换矩阵

$$\boldsymbol{E}_1=\begin{bmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{bmatrix},$$

则

$$\boldsymbol{E}_1\boldsymbol{A}=\begin{bmatrix}6 & 18 & -12\\ 2 & 4 & -2\\ 3 & 17 & 10\end{bmatrix}.$$

由

$$\boldsymbol{M}_1=\begin{bmatrix}1 & 0 & 0\\ -1/3 & 1 & 0\\ -1/2 & 0 & 1\end{bmatrix} \Rightarrow \boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}=\begin{bmatrix}6 & 18 & -12\\ 0 & -2 & 2\\ 0 & 8 & 16\end{bmatrix}.$$

现在为使 $\boldsymbol{M}_2$ 中乘子尽可能小, 我们需要互换第 2 行和第 3 行. 所以, 当

$$\boldsymbol{E}_2=\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 1 & 0\end{bmatrix} \quad \text{和} \quad \boldsymbol{M}_2=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 1/4 & 1\end{bmatrix}$$

时, 有

$$\boldsymbol{M}_2\boldsymbol{E}_2\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}=\begin{bmatrix}6 & 18 & -12\\ 0 & 8 & 16\\ 0 & 0 & 6\end{bmatrix}.$$

此例刻画了行互换的基本思想. 一般情况下, 我们有

**for** $k=1:n-1$

　　确定交换矩阵 $E_k$ 使得 $E_k(1:k,1:k)=I_k$ 且对于 $E_kA$ 的第 $k$ 列 $z$ 有 $|z(k)|=\|z(k:n)\|_\infty$

　　$A=E_kA$

确定高斯变换 $M_k$, 使得 $M_kA$ 的第 $k$ 列 $v$ 满足 $v(k+1:n)=0$

$A=M_kA$

**end**

此特别的对行交换的技巧称为列选主元. 整个过程完成后, 得到 $\boldsymbol{M}_{n-1}\boldsymbol{E}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}=\boldsymbol{U}$, 是一上三角形矩阵.

由于列选主元, 所有乘子的绝对值都不超过 1. 这里由于对 $k=1:n-1$ 都有

$$|(\boldsymbol{E}_k\boldsymbol{M}_{k-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A})_{kk}|=\max_{k\leqslant i\leqslant n}|(\boldsymbol{E}_k\boldsymbol{M}_{k-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A})_{ik}|.$$

所以, 列选主元能有效地保证不出现任意大的乘子.

### 3.4.3 列选主元: 细节

现在, 我们给出带列选主元的高斯消去法整个算法细节.

**算法 3.4.1 (带列选主元的高斯消去法)** 设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$, 则本算法计算高斯变换 $\boldsymbol{M}_1,\cdots\boldsymbol{M}_{n-1}$ 以及交换置换矩阵 $\boldsymbol{E}_1,\cdots,\boldsymbol{E}_{n-1}$ 使得 $\boldsymbol{M}_{n-1}\boldsymbol{E}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}=\boldsymbol{U}$ 是上三角形矩阵. 所有乘子的绝对值都不大于 1. $A(1:k,k)$ 被 $U(1:k,k)$ 所覆盖 $(k=1:n)$. $A(k+1:n,k)$ 被 $-M_k(k+1:n,k)$ 所覆盖 $(k=1:n-1)$. 整数向量 $p(1:n-1)$ 是交换置换指标. 确切地说, 对 $k=1:n-1$, $\boldsymbol{E}_k$ 交换第 $k$ 行和第 $p(k)$ 行.

**for** $k=1:n-1$

  找到 $\mu$ 使得 $k\leqslant\mu\leqslant n$ 且 $|A(\mu,k)|=\|A(k:n,k)\|_\infty$

  $A(k,k:n)\leftrightarrow A(\mu,k:n)$

  $p(k)=\mu$

  **if** $A(k,k)\neq 0$

    rows$=k+1:n$

    $A(\text{rows},k)=A(\text{rows},k)/A(k,k)$

    $A(\text{rows},\text{rows})=A(\text{rows},\text{rows})-A(\text{rows},k)A(k,\text{rows})$

  **end**

**end**

注意到, 如果在第 $k$ 步中 $\|A(k:n,k)\|_\infty=0$, 则在精确运算时 $\boldsymbol{A}$ 的前 $k$ 列是线性相关的. 与算法 3.2.1 不同的是, 在这里没有困难, 我们可简单地跳过零主元.

从浮点运算的角度看, 列选主元所增加的工作量并不大, 因为在选主元时进行比较的次数为 $O(n^2)$. 而整个算法的浮点运算次数为 $\dfrac{2}{3}n^3$.

执行算法 3.4.1 之后, 解线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 需要

- 计算 $\boldsymbol{y}=\boldsymbol{M}_{n-1}\boldsymbol{E}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{b}$
- 求解三角线性方程组 $\boldsymbol{Ux}=\boldsymbol{y}$.

所有需要的信息都储存于 $\boldsymbol{A}$ 和主元向量 $\boldsymbol{p}$ 中. 事实上, 计算过程

**for** $k=1:n-1$

$b(k) \leftrightarrow b(p(k))$

$b(k+1:n) = b(k+1:n) - b(k)A(k+1:n,k)$

**end**

将 $b$ 用 $\boldsymbol{M}_{n-1}\boldsymbol{E}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1$ 覆盖.

**例 3.4.1**　如果算法 3.4.1 用于

$$\boldsymbol{A} = \begin{bmatrix} 3 & 17 & 10 \\ 2 & 4 & -2 \\ 6 & 18 & -12 \end{bmatrix},$$

则结束时有

$$\boldsymbol{A} = \begin{bmatrix} 6 & 18 & -12 \\ 1/3 & 8 & 16 \\ 1/2 & -1/4 & 6 \end{bmatrix}$$

以及 $p = [3\ \ 3]$. 这两个量包含了与以下分解相关的所有信息:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/4 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ -1/3 & 1 & 0 \\ -1/2 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}\boldsymbol{A}$$

$$= \begin{bmatrix} 6 & 18 & -12 \\ 0 & 8 & 16 \\ 0 & 0 & 6 \end{bmatrix}.$$

### 3.4.4　$L$ 储存于何处?

带选主元的高斯消去法计算出 $\boldsymbol{A}$ 经过行置换后的 LU 分解. 此证明仅仅是下标比较.

**定理 3.4.1**　设用带列选主的高斯消去法 (算法 3.4.1) 所求得的上三角形矩阵为

$$\boldsymbol{M}_{n-1}\boldsymbol{E}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A} = \boldsymbol{U}. \tag{3.4.1}$$

则

$$\boldsymbol{PA} = \boldsymbol{LU},$$

其中 $\boldsymbol{P} = \boldsymbol{E}_{n-1}\cdots\boldsymbol{E}_1, \boldsymbol{L}$ 是单位下三角形矩阵且 $|l_{ij}| \leqslant 1$. $\boldsymbol{L}$ 的第 $k$ 列在对角线以下的部分是第 $k$ 个高斯向量的置换形式. 确切地说, 如果 $\boldsymbol{M}_k = \boldsymbol{I} - \boldsymbol{\tau}^{(k)}e_k^T$, 则 $L(k+1:n,k) = g(k+1:n)$, 其中 $g = \boldsymbol{E}_{n-1}\cdots\boldsymbol{E}_{k+1}\boldsymbol{\tau}^{(k)}$.

**证明**　从 (3.4.1) 可知 $\widetilde{\boldsymbol{M}}_{n-1}\cdots\widetilde{\boldsymbol{M}}_1\boldsymbol{PA} = \boldsymbol{U}$, 其中 $\widetilde{\boldsymbol{M}}_{n-1} = \boldsymbol{M}_{n-1}$ 以及

$$\widetilde{\boldsymbol{M}}_k = \boldsymbol{E}_{n-1}\cdots\boldsymbol{E}_{k+1}\boldsymbol{M}_k\boldsymbol{E}_{k+1}\cdots\boldsymbol{E}_{n-1}, \quad k \leqslant n-2.$$

由于 $\boldsymbol{E}_j$ 是关于 $j$ 行与 $\mu$ 行 $(\mu \geqslant j)$ 的互换, 所以有 $E_j(1:j-1,1:j-1) = \boldsymbol{I}_{j-1}$. 从而每个 $\widetilde{\boldsymbol{M}}_k$ 都是高斯变换, 其高斯向量为 $\widetilde{\boldsymbol{\tau}}^{(k)} = \boldsymbol{E}_{n-1}\cdots\boldsymbol{E}_{k+1}\boldsymbol{\tau}^{(k)}$. □

此定理的作用之一是, 让人容易看到该如何修改算法 3.4.1 使得在结束时 $A(i,j)$ 储存 $L(i,j)$ (对 $i>j$). 我们仅需把 $E_k$ 对所有已算出的高斯向量进行变换. 这可在算法 3.4.1 中通过将 "$A(k,k:n) \leftrightarrow A(\mu,k:n)$" 换成 "$A(k,1:n) \leftrightarrow A(\mu,1:n)$" 而实现.

**例 3.4.2** 例 3.4.1 中矩阵之分解 $PA=LU$ 是

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} 3 & 17 & 10 \\ 2 & 4 & -2 \\ 6 & 18 & -12 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/3 & -1/4 & 1 \end{bmatrix}\begin{bmatrix} 6 & 18 & -12 \\ 0 & 8 & 16 \\ 0 & 0 & 6 \end{bmatrix}.$$

### 3.4.5 Gaxpy 形式

在 3.2 节我们给出了计算 LU 分解的外积形式和 gaxpy 形式. 已经讨论了外积形式的选主元法, 很自然将 gaxpy 形式与选主元结合. 回顾一下 (3.2.5) 对一般形式的 gaxpy LU 过程:

$L=I$
$U=0$
**for** $j=1:n$
  **if** $j=1$
    $v(j:n)=A(j:n,j)$
  **else**
    解 $L(1:j-1,1:j-1)z=A(1:j-1,j)$ 求出 $z$
      令 $U(1:j-1,j)=z$
    $v(j:n)=A(j:n,j)-L(j:n,1:j-1)z$
  **end**
  **if** $j<n$
    $L(j+1:n,j)=v(j+1:n)/v(j)$
  **end**
  $U(j,j)=v(j)$
**end**

列选主要求找 $|v(j:n)|$ 的最大元素且相应地进行. 假定 $\boldsymbol{A}$ 非奇异, 从而不会有零主元, 于是我们得到

$L=I; U=0$
**for** $j=1:n$
  **if** $j=1$
    $v(j:n)=A(j:n,j)$
  **else**
    解 $L(1:j-1,1:j-1)z=A(1:j-1,j)$ 求出 $z$

令 $U(1:j-1,j)=z$

$v(j:n)=A(j:n,j)-L(j:n,1:j-1)z$ (3.4.2)

**end**

**if** $j<n$

找出 $\mu$, 满足 $k\leqslant\mu\leqslant n$, 使得 $|v(\mu)|=\|v(j:n)\|_\infty$.

$p(j)=\mu$

$v(j)\leftrightarrow v(\mu)$

$A(j,j+1:n)\leftrightarrow A(\mu,j+1:n)$

$L(j+1:n,j)=v(j+1:n)/v(j)$

**if** $j>1$

$L(j,1:j-1)\leftrightarrow L(\mu,1:j-1)$

**end**

**end**

$U(j,j)=v(j)$

**end**

在此程序中, 我们得到了分解 $\boldsymbol{PA}=\boldsymbol{LU}$, 其中 $\boldsymbol{P}=\boldsymbol{E}_{n-1}\cdots\boldsymbol{E}_1,\boldsymbol{E}_k$ 是互换 $n\times n$ 单位矩阵的第 $k$ 行和第 $p(k)$ 行的交换矩阵. 和算法 3.4.1 一样, 此算法需要要 $\frac{2}{3}n^3$ 个 flop 和 $O(n^2)$ 次比较.

### 3.4.6 误差分析

现在我们考查列选主所得到的稳定性. 这需要考虑消去时以及解三角方程组时的舍入误差. 记住在置换运算中没有舍入误差, 从定理 2.3.2 不难证明计算出来的 $\hat{x}$ 满足 $(\boldsymbol{A}+\boldsymbol{E})\hat{x}=b$, 其中

$$|\boldsymbol{E}|\leqslant n\mathbf{u}(3|\boldsymbol{A}|+5\hat{\boldsymbol{P}}^{\mathrm{T}}|\hat{\boldsymbol{L}}||\hat{\boldsymbol{U}}|)+O(\mathbf{u}^2). \tag{3.4.3}$$

这是我们假定 $\hat{\boldsymbol{P}},\hat{\boldsymbol{L}}$ 和 $\hat{\boldsymbol{U}}$ 是以上算法产生的 $\boldsymbol{P},\boldsymbol{L}$ 和 $\boldsymbol{U}$ 的计算形式. 选主元保证 $\hat{\boldsymbol{L}}$ 的元素不超过 1. 所以 $\|\hat{\boldsymbol{L}}\|_\infty\leqslant n$. 因而我们有界

$$\|\boldsymbol{E}\|_\infty\leqslant n\mathbf{u}(3\|\boldsymbol{A}\|_\infty+5n\|\hat{\boldsymbol{U}}\|_\infty)+O(\mathbf{u}^2). \tag{3.4.4}$$

下面的问题是估计 $\|\hat{\boldsymbol{U}}\|_\infty$. 定义**增长因子** $\rho$:

$$\rho=\max_{i,j,k}\frac{|\hat{a}_{ij}^{(k)}|}{\|\boldsymbol{A}\|_\infty}, \tag{3.4.5}$$

这里 $\hat{\boldsymbol{A}}^{(k)}$ 是矩阵 $\boldsymbol{A}^{(h)}=\boldsymbol{M}_k\boldsymbol{E}_k\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}$ 的计算值. 则有

$$\|\boldsymbol{E}\|_\infty\leqslant 8n^3\rho\|\boldsymbol{A}\|_\infty\mathbf{u}+O(\mathbf{u}^2). \tag{3.4.6}$$

此界能否与理想界 (3.3.1) 相比取决于增长因子 $\rho$ 的大小. (因子 $n^3$ 在实际中不太重要, 在此讨论中不予考虑.) 此增长因子指出在消去过程中元素能有多大. 在实际

中, $\rho$ 的量级常常是 10, 但它也可能大到 $2^{n-1}$. 即便如此, 大多数数值分析专家认为, 实际的带列选主元的高斯消去法中元素讯速增长的情况是非常罕见的. 所以该方法可以放心使用.

**例 3.4.3** 设带列选主元的高斯消去法用于问题

$$\begin{bmatrix} 0.001 & 1.00 \\ 1.00 & 2.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ 3.00 \end{bmatrix}$$

假设浮点运算的 $\beta = 10, t = 3$, 则

$$\boldsymbol{P} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \hat{\boldsymbol{L}} = \begin{bmatrix} 1.00 & 0 \\ 0.001 & 1.00 \end{bmatrix}, \quad \hat{\boldsymbol{U}} = \begin{bmatrix} 1.00 & 2.00 \\ 0 & 1.00 \end{bmatrix}$$

以及 $\hat{x} = (1.00, 0.996)^{\mathrm{T}}$. 试比较例 3.3.1.

**例 3.4.4** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 之定义为

$$a_{ij} = \begin{cases} 1, & \text{如果 } i = j \text{ 或 } j = n, \\ -1, & \text{如果 } i > j, \\ 0, & \text{其他}. \end{cases}$$

则 $\boldsymbol{A}$ 的 LU 分解满足 $|l_{ij}| \leqslant 1$ 且 $\boldsymbol{u}_{nn} = 2^{n-1}$.

### 3.4.7 分块高斯消去

带列选主元的高斯消去法可整理成主要为 3 级运算. 我们给出分块外积的详细过程, 分块 gaxpy 和分块点积也同样是可能的. 见 Duyde and Duff(1988).

设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 为清楚起见假定 $n = rN$. 将 $\boldsymbol{A}$ 划分为

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$$
$$\qquad\quad r \qquad n-r$$

分块消去的第一步是典型的. 它的做法如下:

- 用标量的带列选主的高斯消去法 (即算法 3.4.1 的长方形式) 计算置换阵 $\boldsymbol{P}_1 \in \mathbb{R}^{n\times n}$, 单位下三角形矩阵 $\boldsymbol{L}_{11} \in \mathbb{R}^{r\times r}$ 和上三角形矩阵 $\boldsymbol{U}_{11} \in \mathbb{R}^{r\times r}$ 使得

$$\boldsymbol{P}_1 \begin{bmatrix} \boldsymbol{A}_{11} \\ \boldsymbol{A}_{21} \end{bmatrix} = \begin{bmatrix} \boldsymbol{L}_{11} \\ \boldsymbol{L}_{21} \end{bmatrix} \boldsymbol{U}_{11}.$$

- 将 $\boldsymbol{P}_1$ 作用于 $\boldsymbol{A}$ 的其他部分

$$\begin{bmatrix} \widetilde{\boldsymbol{A}}_{12} \\ \widetilde{\boldsymbol{A}}_{22} \end{bmatrix} = \boldsymbol{P}_1 \begin{bmatrix} \boldsymbol{A}_{12} \\ \boldsymbol{A}_{22} \end{bmatrix}.$$

- 求多右端项的下三角问题

$$\boldsymbol{L}_{11}\boldsymbol{U}_{12} = \widetilde{\boldsymbol{A}}_{12}$$

- 进行 3 级修正

$$\widetilde{A} = \widetilde{A}_{22} - L_{21}U_{12}.$$

利用这些计算我们得到分解

$$P_1A = \begin{bmatrix} L_{11} & 0 \\ L_{21} & I_{n-r} \end{bmatrix} \begin{bmatrix} I_r & 0 \\ 0 & \widetilde{A} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & I_{n-r} \end{bmatrix}.$$

然后, 以上做法可重新用于 $\widetilde{A}$ 的前 $r$ 列.

一般地, 分块算法的第 $k$ 步 $(1 \leqslant k \leqslant N-1)$ 对一个 $(n-(k-1)r) \times r$ 矩阵做标量的高斯消去法. 求解一个 $r \times (n-kr)$ 的多右端项问题以及进行维数为 $(n-kr) \times (n-kr)$ 的 3 级修正. 整个算法的 3 级比例大致为 $1 - \frac{3}{2N}$. 因此, 此方法在 $N$ 很大时主要是矩阵乘法.

### 3.4.8 全选主元

另一个选主元策略称为全选主元法, 它具有其对应增长因子界比 $2^{n-1}$ 小得多的性质. 我们重述, 在列选主中第 $k$ 个主元是从当前列的一部分 $A(k:n,k)$ 中寻找. 在全选主元法中, 是把当前子矩阵 $A(k:n,k:n)$ 中的最大元素置换到 $(k,k)$ 位置. 这样我们求得上三角分解: $M_{n-1}E_{n-1}\cdots M_1E_1AF_1\cdots F_{n-1} = U$, 在第 $k$ 步时, 要处理的矩阵是

$$A^{(k-1)} = M_{k-1}E_{k-1}\cdots M_1E_1AF_1\cdots F_{k-1},$$

而且需要确定置换矩阵 $E_k$ 和 $F_k$ 使得

$$|(E_kA^{(k-1)}F_k)_{kk}| = \max_{k\leqslant i,j\leqslant n} |(E_kA^{(k-1)}F_k)_{ij}|.$$

类似于定理 3.4.1, 我们有以下结论.

**定理 3.4.2** 设带全选主元的高斯消去法用来计算上三角分解

$$M_{n-1}E_{n-1}\cdots M_1E_1AF_1\cdots F_{n-1} = U, \tag{3.4.7}$$

则有

$$PAQ = LU,$$

其中 $P = E_{n-1}\cdots E_1, Q = F_1\cdots F_{n-1}$, $L$ 是满足于 $|l_{ij}| \leqslant 1$ 的单位下三角形矩阵. $L$ 的第 $k$ 列在对角线以下的部分是第 $k$ 个高斯向量的置换形式. 确切地说, 如果 $M_k = I - \tau^{(k)}e_k^{\mathrm{T}}$, 则 $L(k+1:n,k) = g(k+1:n)$, 其中 $g = E_{n-1}\cdots E_{k+1}\tau^{(k)}$.

**证明** 此证明与定理 3.4.1 之证明类似. 详细证明留给读者. □

以下是带全选主元的高斯消去法的细节.

**算法 3.4.2 (带全选主元的高斯消去法)** 本算法计算全选主元分解 $PAQ = LU$, 其中 $L$ 是单位下三角形矩阵, $U$ 是上三角形矩阵. $P = E_{n-1}\cdots E_1, Q = F_1\cdots F_{n-1}$ 是交换置换矩阵之乘积. $A(1:k,k)$ 由 $U(1:k,k)$ 所覆盖 $(k=1:n)$.

$A(k+1:n,k)$ 被 $L(k+1:n,k)$ 所覆盖 $(k=1:n-1)$. $\boldsymbol{E}_k$ 交换第 $k$ 行和第 $p(k)$ 行, $F_k$ 交换第 $k$ 列和第 $q(k)$ 列.

**for** $k=1:n-1$

　　确定满足 $k\leqslant\mu\leqslant n$ 和 $k\leqslant\lambda\leqslant n$ 的 $\mu$ 和 $\lambda$ 使得

$$|A(\mu,\lambda)|=\max\{|A(i,j)|:i=k:n,j=k:n\}$$

　　$A(k,1:n)\leftrightarrow A(\mu,1:n)$

　　$A(1:n,k)\leftrightarrow A(1:n,\lambda)$

　　$p(k)=\mu$

　　$q(k)=\lambda$

　　**if** $A(k,k)\neq 0$

　　　　rows$=k+1:n$

　　　　$A(\text{rows},k)=A(\text{rows},k)/A(k,k)$

　　　　$A(\text{rows},\text{rows})=A(\text{rows},\text{rows})-A(\text{rows},k)\cdot A(k,\text{rows})$

　　**end**

**end**

此算法需要 $2n^3/3$ 个 flop 和 $O(n^3)$ 次比较. 与列选主不同, 全选主元法由于每步的两维数组搜索需要增加很大的选主元工作量.

### 3.4.9 关于全选主的说明

假定 $\text{rank}(\boldsymbol{A})=r<n$, 则在 $r+1$ 步之前 $A(r+1:n,r+1:n)=0$. 这意味着 $\boldsymbol{E}_k=\boldsymbol{F}_k=\boldsymbol{M}_k=\boldsymbol{I}$ $(k=r+1:n)$. 所以算法可在 $r$ 步之后结束, 所得到的分解为

$$\boldsymbol{PAQ}=\boldsymbol{LU}=\begin{bmatrix}\boldsymbol{L}_{11} & \boldsymbol{0}\\ \boldsymbol{L}_{21} & \boldsymbol{I}_{n-r}\end{bmatrix}\begin{bmatrix}\boldsymbol{U}_{11} & \boldsymbol{U}_{12}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}.$$

这里 $\boldsymbol{L}_{11}$ 和 $\boldsymbol{U}_{11}$ 是 $r\times r$, $\boldsymbol{L}_{21}$ 和 $\boldsymbol{U}_{12}^{\mathrm{T}}$ 是 $(n-r)\times r$. 因此, 带全选主元的高斯消去法原则上可用来确定矩阵的秩. 但是, 舍入误差使得碰到精确的零主元不太可能. 在实际中, 如果在 $k+1$ 步中主元是充分小, 则我们可以 "断言" $\boldsymbol{A}$ 的秩为 $r$. 决定数值秩的问题将在 5.4 节中详细讨论.

Wilklhson(1961) 证明了在精确运算下矩阵 $\boldsymbol{A}^{(k)}=\boldsymbol{M}_k\boldsymbol{E}_k\cdots\boldsymbol{M}_1\boldsymbol{E}_1\boldsymbol{A}\boldsymbol{E}_1\cdots\boldsymbol{F}_k$ 的元素满足

$$|a_{ij}^{(k)}|\leqslant k^{\frac{1}{2}}(2.3^{1/2}\cdots k^{1/(k-1)})^{1/2}\max|a_{ij}|, \tag{3.4.8}$$

此上界是关于 $k$ 较慢增长的函数. 该结果以及大量的实例表明 $\rho$ 总是不太大 (如 $\rho=10$), 这让我们推断带全选主元的高斯消去法是稳定的. 该方法在 (3.3.1) 的意义下精确求解一个附近的线性方程组 $(\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{x}}=\boldsymbol{b}$. 但是, 在实际中, 除了需要确定矩阵的秩, 看不出有什么理由选择全选主元而不是选择列选主元法.

**例 3.4.5**　如果浮点运算 $\beta=10,t=3$ 和带全选主元的高斯消去法应用于

$$\begin{bmatrix} 0.001 & 1.00 \\ 1.00 & 2.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ 3.00 \end{bmatrix}.$$

则

$$\boldsymbol{P} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{Q} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

$$\hat{\boldsymbol{L}} = \begin{bmatrix} 1.00 & 0.00 \\ 0.50 & 1.00 \end{bmatrix}, \quad \hat{\boldsymbol{U}} = \begin{bmatrix} 2.00 & 1.00 \\ 0.00 & 0.499 \end{bmatrix}$$

且 $\hat{x} = (1.00, 1.00)^{\mathrm{T}}$. 将其与例 3.3.1 和例 3.4.3 比较.

### 3.4.10 不必选主元

在某些特定情况不必选主元. 指明这些情形是重要的, 因为选主元常会影响方法的效率. 为举例说明选主元可安全地略去, 我们考虑对角占优矩阵. 如果

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \quad i = 1:n,$$

则我们称 $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ 是严格对角占优的. 以下定理表明此性质可保证很好的不用选主元的 LU 分解.

**定理 3.4.3** 如果 $\boldsymbol{A}^{\mathrm{T}}$ 是严格对角占优的, 则 $\boldsymbol{A}$ 有 LU 分解且 $|l_{ij}| \leqslant 1$. 换句话说, 如果应用算法 3.4.1, 则 $\boldsymbol{P} = \boldsymbol{I}$.

**证明** 将 $\boldsymbol{A}$ 划分为

$$\boldsymbol{A} = \begin{bmatrix} \alpha & \boldsymbol{w}^{\mathrm{T}} \\ \boldsymbol{v} & \boldsymbol{C} \end{bmatrix},$$

其中 $\alpha$ 是 $1 \times 1$ 的. 注意到应用一步外积形式的 LU 分解后我们有

$$\begin{bmatrix} \alpha & \boldsymbol{w}^{\mathrm{T}} \\ \boldsymbol{v} & \boldsymbol{C} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \boldsymbol{v}/\alpha & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \boldsymbol{C} - \boldsymbol{v}\boldsymbol{w}^{\mathrm{T}}/\alpha \end{bmatrix} \begin{bmatrix} \alpha & \boldsymbol{w}^{\mathrm{T}} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}.$$

如果我们可证 $\boldsymbol{B} = \boldsymbol{C} - \boldsymbol{v}\boldsymbol{w}^{\mathrm{T}}/\alpha$ 的转置是严格对角占优的, 则对 $n$ 进行归纳法可得到定理. 这是由于我们可假定 $\boldsymbol{B}$ 的 LU 分解为 $\boldsymbol{B} = \boldsymbol{L}_1\boldsymbol{U}_1$, 即得

$$\boldsymbol{A} = \begin{bmatrix} 1 & \boldsymbol{0} \\ \boldsymbol{v}/\alpha & \boldsymbol{L}_1 \end{bmatrix} \begin{bmatrix} \alpha & \boldsymbol{w}^{\mathrm{T}} \\ \boldsymbol{0} & \boldsymbol{U}_1 \end{bmatrix} \equiv \boldsymbol{L}\boldsymbol{U}.$$

但是, 证明 $\boldsymbol{B}^{\mathrm{T}}$ 是严格对角占优的是显而易见的. 从定义我们有

$$\begin{aligned} \sum_{\substack{i=1 \\ i \neq j}}^{n-1} |b_{ij}| = \sum_{\substack{i=1 \\ i \neq j}}^{n-1} |c_{ij} - v_i w_j/\alpha| &\leqslant \sum_{\substack{i=1 \\ i \neq j}}^{n-1} |c_{ij}| + \frac{|w_j|}{|\alpha|} \sum_{\substack{i=1 \\ i \neq j}}^{n-1} |v_i| \\ &\leqslant (|c_{jj}| - |w_j|) + \frac{|w_j|}{|\alpha|}(|\alpha| - |v_j|) \\ &\leqslant \left| c_{jj} - \frac{w_j v_j}{\alpha} \right| = |b_{jj}|. \end{aligned}$$

□

### 3.4.11 一些应用

我们以一些例子结束本节，这些例子表明对于不同线性方程如何用矩阵分解来考虑.

假定 $\boldsymbol{A}$ 是非奇异的且是 $n\times n$ 的, $\boldsymbol{B}$ 是 $n\times p$ 的. 考虑问题：找 $\boldsymbol{X}(n\times p)$ 使得 $\boldsymbol{AX}=\boldsymbol{B}$, 即多右端项问题. 如果 $\boldsymbol{X}=[x_1,\cdots,x_p]$ 和 $\boldsymbol{B}=[b_1,\cdots,b_p]$ 是列划分, 则

计算 $PA=LU$
**for** $k=1:p$
  解 $Ly=Pb_k$ (3.4.9)
  解 $Ux_k=y$
**end**

注意 $\boldsymbol{A}$ 仅需分解一次. 如果 $\boldsymbol{B}=\boldsymbol{I}_n$, 则我们得到了 $\boldsymbol{A}^{-1}$.

另一个例子是 "在循环外" 得到 LU 分解. 假定我们需要求解线性方程组 $\boldsymbol{A}^k\boldsymbol{x}=\boldsymbol{b}$, 其中 $\boldsymbol{A}\in\mathbb{R}^{n\times n},\boldsymbol{b}\in\mathbb{R}^n,k$ 是正整数. 一种方式是先求出 $\boldsymbol{C}=\boldsymbol{A}^k$ 然后求解 $\boldsymbol{Cx}=\boldsymbol{b}$. 但是矩阵乘法可完全避免:

计算 $\boldsymbol{PA}=\boldsymbol{LU}$
**for** $j=1:k$
  用 $Ly=Pb$ 之解覆盖 $b$ (3.4.10)
  用 $Ux=b$ 之解覆盖 $b$.
**end**

最后我们说明如何避免陷入显式计算逆矩阵的陷阱. 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$, $\boldsymbol{d}\in\mathbb{R}^n$ 和 $\boldsymbol{c}\in\mathbb{R}^n$, 我们要计算 $s=\boldsymbol{c}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{d}$. 一种方式是如上所建议的计算 $\boldsymbol{X}=\boldsymbol{A}^{-1}$, 然后求 $s=\boldsymbol{c}^{\mathrm{T}}\boldsymbol{Xd}$. 更有效的方法是计算 $\boldsymbol{PA}=\boldsymbol{LU}$ 然后解三角方程组 $\boldsymbol{Ly}=\boldsymbol{Pd}$ 和 $\boldsymbol{Ux}=\boldsymbol{y}$. 其结果是 $s=\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}$. 此例子所强调指出的是, 当一个公式中有逆矩阵时, 我们应该从解线性方程组的角度来考虑而不是显式求逆.

## 习 题

**3.4.1** 令 $\boldsymbol{A}=\boldsymbol{LU}$ 是 $n\times n$ 矩阵 $\boldsymbol{A}$ 的 LU 分解且 $|l_{ij}|\leqslant 1$. 设 $\boldsymbol{a}_i^{\mathrm{T}}$ 和 $\boldsymbol{u}_i^{\mathrm{T}}$ 分别表示 $\boldsymbol{A}$ 和 $\boldsymbol{U}$ 的第 $i$ 行. 证明

$$\boldsymbol{u}_i^{\mathrm{T}}=\boldsymbol{a}_i^{\mathrm{T}}-\sum_{j=1}^{i-1}l_{ij}\boldsymbol{u}_j^{\mathrm{T}}$$

并且用它证明 $\|\boldsymbol{U}\|_\infty\leqslant 2^{n-1}\|\boldsymbol{A}\|_\infty$. (提示：取模且用归纳法.)

**3.4.2** 证明如果 $\boldsymbol{PAQ}=\boldsymbol{LU}$ 是由带全选主元的高斯消去法所求得, 则 $U(i,i:n)$ 中任何元素的绝对值都不大于 $|u_{ii}|$.

**3.4.3** 假设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 有 LU 分解且 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 已知. 给出一个大约 $(n-j)^2+(n-i)^2$ 个 flop 的计算 $\boldsymbol{A}^{-1}$ 中 $(i,j)$ 元素的算法.

**3.4.4** 假设 $\hat{X}$ 是通过 (3.4.9) 所求出的数值逆. 给出 $\|\boldsymbol{A}\hat{\boldsymbol{X}}-\boldsymbol{I}\|_F$ 之上界.

**3.4.5** 证明定理 3.4.2.

**3.4.6** 推广算法 3.4.3 使其可分解任何长方矩阵.

**3.4.7** 给出 3.4.7 中所简述的分块消去法之详细形式.

## 本节注释与参考文献

算法 3.4.1 的 Agol 形式由下文给出.

H. J. Bowdler, R. S. Martin, G. Peters, and J. H. Wilkinson (1966). "Solution of Real and Complex Systems of Linear Equations," *Numer. Math.* 8, 217–234. 也见 Wilkinson and Reinsch (1971, 93–110).

当全选主元时, 猜想 $|a_{ij}^{(k)}| \leqslant n \max |a_{ij}|$ 在 $n = 4$ 的情形下被下文证明:

C. W. Cryer (1968). "Pivot Size in Gaussian Elimination," *Numer. Math.* 12, 335–345.

关于元素增长以及选主元的其他文章包括:

J. K. Reid (1971). "A. Note on the Stability of Gaussian Elimination," *J. Inst. Math. Applics.* 8, 374–375.

P. A. Businger (1971). "Monitoring the Numerical Stability of Gaussian Elimination," *Numer. Math.* 16, 360–361.

A. M. Cohen (1974). "A Note on Pivot Size in Gaussian Elimination," *Lin. Alg. and Its Applic.* 8, 361–368.

A. M. Erisman and J. K. Reid (1974). "Monitoring the Stability of the Triangular Factorization of a Sparse Matrix," *Numer. Math.* 22, 183–186.

J. Day and B. Peterson (1988). "Growth in Gaussian Elimination," *Amer. Math. Monthly* 95, 489–513.

N. J. Higham and D. J. Higham (1989). "Large Growth Factors in Gaussian Elimination with Pivoting," *SIAM J. Matrix Anal. Appl.* 10, 155–164.

L. N. Trefethen and R. S. Schreiber (1990). "Average-Case Stability of Gaussian Elimination," *SIAM J. Matrix Anal. Appl.* 11, 335–360.

N. Gould (1991). "On Growth in Gaussian Elimination with Complete Pivoting," *SIAM J. Matrix Anal. Appl.* 12, 354–361.

A. Edelman (1992). "The Complete Pivoting Conjecture for Gaussian Elimination is False," *The Mathematica Journal* 2, 58–61.

S. J. Wright (1993). "A Collection of Problems for Which Gaussian Elimination with Partial Pivoting is Unstable," *SIAM J. Sci. and Stat. Comp.* 14, 231–238.

L. V. Foster (1994). "Gaussian Elimination with Partial Pivoting Can Fail in Practice," *SIAM J. Matrix Anal. Appl.* 15, 1354–1362.

A. Edelman and W. Mascarenhas (1995). "On the Complete Pivoting Conjecture for a Hadamard Matrix of Order 12," *Linear and Multilinear Algebra* 38, 181–185.

在元素增长的讨论中构造稀疏高斯消去法程序是有趣的. 这是因为为了尽可能少地填充稍大的乘子, 有时是可接受的. 见:

I. S. Duff, A. M. Erisman, and J. K. Reid (1986). *Direct Methods for Sparse Matrices*, Oxford University Press.

关于小主元与接近奇异的关系可见参考文献:

T. F. Chan (1985). "On the Existence and Computation of LU Factorizations with small pivots," *Math. Comp.* 42, 535–548.

我们没讨论的一种选主元策略是成对选主元. 在该方法中, 一个 $2\times 2$ 高斯变换用来把 $A$ 的下三角部分消为零. 此技巧在某些特定的多处理计算机上很适合. 因为每步只是相邻之行的组合. 见:

D. Sorensen (1985). "Analysis of Pairwise Pivoting in Gaussian Elimination," *IEEE Trans. on Computers* C-34, 274–278.

关于确定一类矩阵不必选主元的范例文章. 可见:

S. Serbin (1980). "On Factoring a Class of Complex Symmetric Matrices Without Pivoting," *Math. Comp.* 35, 1231–1234.

正如标量高斯消去法有六种 "标准" 形式, 分块的高斯消去法也有六种标准形式. 关于这些方法及实现之讨论可见:

K. Gallivan, W. Jalby, U. Meier, and A. H. Sameh (1988). "Impact of Hierarchical Memory Systems on Linear Algebra Algorithm Design," *Int'l J. Supercomputer Applic.*2, 12–48.

## 3.5 改进与精度估计

假定用带列选主元的高斯消去法求解 $n\times n$ 线性方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$. 设所用的浮点运算是 $t$ 位, 基是 $\beta$. 公式 (3.4.6) 就是说, 如果增长因子适中, 则计算解 $\hat{\boldsymbol{x}}$ 满足

$$(\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{x}}=\boldsymbol{b},\quad \|\boldsymbol{E}\|_\infty\approx \mathbf{u}\|\boldsymbol{A}\|_\infty,\quad \mathbf{u}=\frac{1}{2}\beta^{-t}. \tag{3.5.1}$$

在本节, 我们考查此结果的实际含意. 首先我们强调有必要区分余量和精度. 然后讨论加权、迭代改进和条件数估计. 关于这些方向的详细讨论可见 Higham(1996).

我们先做两点记号的说明. 第一, 整节中用到的都是无穷范数, 因为它在舍入误差分析以及实际误差估计中十分方便. 第二, 我们在本节提到 "高斯消去法" 时是指带某种稳定化主元技巧 (如列选主元) 的高斯消去法.

### 3.5.1 余量与精度

线性方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 之计算解 $\hat{\boldsymbol{x}}$ 的余量是向量 $\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}$. 小的余量意味着 $\boldsymbol{A}\hat{\boldsymbol{x}}$ 能有效地 "预测" 右端项 $\boldsymbol{b}$. 从 (3.5.1) 我们有 $\|\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}\|_\infty\approx\mathbf{u}\|\boldsymbol{A}\|_\infty\|\hat{\boldsymbol{x}}\|_\infty$, 所以得到下列启示.

**启示一** 高斯消去法产生的解 $\hat{x}$ 之余量是比较小的.

小余量并不意味着高精度. 结合 (3.3.2) 和 (3.5.1), 得到

$$\frac{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_\infty} \approx \mathbf{u}\kappa_\infty(\boldsymbol{A}). \tag{3.5.2}$$

这证明了第二个指导原则.

**启示二** 如果单位舍入误差和条件数满足 $\mathbf{u} \approx 10^{-d}$ 和 $\kappa_\infty(\boldsymbol{A}) \approx 10^q$. 则高斯消去法产生的解 $\hat{\boldsymbol{x}}$ 大约有 $d-q$ 位十进制有效数字.

如果 $\mathbf{u}\kappa_\infty(\boldsymbol{A})$ 很大, 则称 $\boldsymbol{A}$ 相对机器精度是病态的.

为说明启示一和启示二, 考虑方程组

$$\begin{bmatrix} 0.986 & 0.579 \\ 0.409 & 0.237 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.235 \\ 0.107 \end{bmatrix},$$

其条件数 $\kappa_\infty(\boldsymbol{A}) \approx 700$, 解 $\boldsymbol{x} = (2, -3)^{\mathrm{T}}$. 以下是不同机器精度下的结果.

| $\beta$ | $t$ | $\hat{x}_1$ | $\hat{x}_2$ | $\dfrac{\Vert\hat{\boldsymbol{x}}-\boldsymbol{x}\Vert_\infty}{\Vert\boldsymbol{x}\Vert_\infty}$ | $\dfrac{\Vert\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}\Vert_\infty}{\Vert\boldsymbol{A}\Vert_\infty\Vert\hat{\boldsymbol{x}}\Vert_\infty}$ |
|---|---|---|---|---|---|
| 10 | 3 | 2.11 | −3.17 | $5\times10^{-2}$ | $2.0\times10^{-3}$ |
| 10 | 4 | 1.986 | −2.975 | $8\times10^{-3}$ | $1.5\times10^{-4}$ |
| 10 | 5 | 2.001 9 | −3.003 2 | $1\times10^{-3}$ | $2.1\times10^{-6}$ |
| 10 | 6 | 2.000 25 | −3.000 94 | $3\times10^{-4}$ | $4.2\times10^{-7}$ |

是否对计算解 $\hat{\boldsymbol{x}}$ 满意取决于原始问题之需求. 在许多应用中, 精度并不重要但小余量却很关键. 在这些情形下, 高斯消去法求出的解 $\hat{\boldsymbol{x}}$ 也许就足够了. 另一方面, 当 $\hat{\boldsymbol{x}}$ 中有效数字之个数有关紧要时, 问题就更复杂, 本节剩下的讨论都与之有关.

### 3.5.2 加权

设 $\beta$ 是机器的进制基, 定义对角矩阵 $\boldsymbol{D}_1$ 和 $\boldsymbol{D}_2$:

$$\boldsymbol{D}_1 = \mathrm{diag}(\beta^{r_1}, \cdots, \beta^{r_n}),$$

$$\boldsymbol{D}_2 = \mathrm{diag}(\beta^{c_1}, \cdots, \beta^{c_n}).$$

$n\times n$ 线性方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 之解可通过用高斯消去法求解**加权方程组** $(\boldsymbol{D}_1^{-1}\boldsymbol{A}\boldsymbol{D}_2)\boldsymbol{y} = \boldsymbol{D}_1^{-1}\boldsymbol{b}$ 然后令 $\boldsymbol{x} = \boldsymbol{D}_2\boldsymbol{y}$ 得到. $\boldsymbol{A}, \boldsymbol{b}$ 和 $\boldsymbol{y}$ 的加权仅需 $O(n^2)$ 个 flop 而且没有舍入误差. 注意到 $\boldsymbol{D}_1$ 加权方程而 $\boldsymbol{D}_2$ 加权未知量.

从启示二知. 若 $\hat{\boldsymbol{x}}$ 和 $\hat{\boldsymbol{y}}$ 是 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 的计算值, 则

$$\frac{\|\boldsymbol{D}_2^{-1}(\hat{\boldsymbol{x}}-\boldsymbol{x})\|_\infty}{\|\boldsymbol{D}_2^{-1}\boldsymbol{x}\|_\infty} = \frac{\|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_\infty}{\|\boldsymbol{y}\|_\infty} \approx \mathbf{u}\kappa_\infty(\boldsymbol{D}_1^{-1}\boldsymbol{A}\boldsymbol{D}_2). \tag{3.5.3}$$

于是, 当能使 $\kappa_\infty(\boldsymbol{D}_1^{-1}\boldsymbol{A}\boldsymbol{D}_2)$ 远小于 $\kappa_\infty(\boldsymbol{A})$ 时, 我们可得到相对更精确的 $\hat{\boldsymbol{x}}$, 只要误差是用 "$\boldsymbol{D}_2$" 范数 $\|\boldsymbol{z}\|_{\boldsymbol{D}_2} = \|\boldsymbol{D}_2^{-1}\boldsymbol{z}\|_\infty$ 定义的. 这就是加权的目的. 注意到这包含了两个问题, 其一是加权问题的条件数, 其二是在 $\boldsymbol{D}_2$ 范数下评价误差的适当性.

一个有意义但十分困难的数学问题是对一般对角矩阵 $\boldsymbol{D}_i$ 和不同 $p$ 求 $\kappa_p(\boldsymbol{D}_1^{-1}\boldsymbol{A}\boldsymbol{D}_2)$ 的精确极小值. 这方面的结果在实际中没有什么用处. 但是, 这并不让人失

望, 因为 (3.5.3) 是一个大致估计式, 而精确地极小化一个近似界没什么意义. 我们需要的是改进计算解 $\hat{x}$ 的精度的快速近似算法.

这一变换的特殊情形是*简单行加权*. 在此方法中 $D_2$ 是单位矩阵而且选择 $D_1$, 使得 $D_1^{-1}A$ 每行大约有相同的无穷范数. 行加权降低了在消去法中把一个很小的数加到一个很大的数的可能性, 这种情形严重损失精度.

比简单行加权稍微复杂的是*行–列平衡*. 其目的是选择 $D_1$ 和 $D_2$ 使得 $D_1^{-1}AD_2$ 的每一行和每一列的无穷范数都属于区间 $[1/\beta, 1]$, 其中 $\beta$ 是浮点系统的基. 关于这方面的工作可见 McKeeman (1962).

对于简单行加权和行–列平衡没 "解决" 加权问题这一点不必过分强调. 事实上, 如果没有加权每种方法都可能得到一个更差的 $\hat{x}$. 关于这一点的详细讨论可见 Forsythe and Moler(1967, 第 11 章). 根本性的建议是对方程和未知量的加权必须针对不同问题进行. 通用性的加权技巧是不可靠的. 最好是基于原始问题描述的每个 $a_{ij}$ 之重要性来进行加权 (如果需要加权的话). 度量单位以及数据误差也应考虑.

**例 3.5.1 (Forsythe and Moler (1967), 第 34, 40 页)** *如果*

$$\begin{bmatrix} 10 & 100\,000 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} 100\,000 \\ 2 \end{bmatrix}.$$

*以及其等价的行加权问题*

$$\begin{bmatrix} 0.0001 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

*都用* $\beta = 10, t = 3$ *的浮点运算求解, 则计算解分别为* $\hat{x} = (0.00, 1.00)^{\mathrm{T}}$ *和* $\hat{x} = (1.00, 1.00)^{\mathrm{T}}$. *注意到* $x = (1.001\cdots, 0.999\cdots)^{\mathrm{T}}$ *是精确解.*

### 3.5.3 迭代改进

设 $Ax = b$ 通过列选主元法 $PA = LU$ 已求解, 假定我们需要改进计算解 $\hat{x}$ 的精度. 如果我们执行

$$\begin{aligned} & r = b - A\hat{x} \\ & \text{解 } Ly = Pr \\ & \text{解 } Uz = y \\ & x_{\text{new}} = \hat{x} + z \end{aligned} \tag{3.5.4}$$

则在精确计算下, $Ax_{\text{new}} = A\hat{x} + Az = (b - r) + r = b$. 不幸的是, 这些步骤的浮点运算所得到的 $x_{\text{new}}$ 不会比 $\hat{x}$ 更精确. 这一点是在意料之中的, 这是由于 $\hat{r} = \text{fl}(b - A\hat{x})$ 如果有有效数字的话, 也只有很少几位有效数字. (回忆启示一.) 因而 $z = \text{fl}(A^{-1}r) \approx A^{-1} \cdot \text{noise} \approx \text{noise}$ 从改进 $\hat{x}$ 精度的角度来说是一个十分差的修改. 但是, Skee(1980) 给出了误差分析, 它表明, 从向后误差的角度看, (3.5.4) 何时可给出一个改进的 $x_{\text{new}}$. 确切地说, 当

$$\tau = (\||A||A^{-1}|\|_\infty)(\max_i(|A||x|)_i / \min_i(|A||x|)_i)$$

不是太大时, 则 (3.5.4) 给出 $\boldsymbol{x}_{\text{new}}$ 使得对非常小的 $\boldsymbol{E}$ 有 $(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}_{\text{new}}=\boldsymbol{b}$. 当然, 当带列选主元的高斯消去法用来求解时, 计算解 $\hat{x}$ 满足一个附近的方程. 但是这对某些保持稀疏性的选主元技巧来说并不一定成立. 在此情形下, 固定精度迭代改进 (3.5.4) 是十分值得的, 也是经济的. 见 Arioli, Demmel, and Duff(1988).

要使 (3.5.4) 给出更精确的 $\boldsymbol{x}$, 有必要用扩充精度的浮点运算来计算余量 $b-\boldsymbol{A}\hat{x}$. 典型地, 如果 $t$ 位数运算用来计算 $\boldsymbol{PA}=\boldsymbol{LU},\boldsymbol{x},\boldsymbol{y}$ 和 $\boldsymbol{z}$, 则用 $2t$ 位数运算, 即双精度来计算 $\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}$. 这一过程可以迭代, 也就是说, 一旦我们计算了 $\boldsymbol{PA}=\boldsymbol{LU}$ 和初始化 $\boldsymbol{x}=\boldsymbol{0}$, 我们可重复如下过程:

$$\begin{aligned}
&r=b-Ax\ (\text{双精度})\\
&\text{解 } Ly=Pr \text{ 得 } y.\\
&\text{解 } Uz=y \text{ 得 } z.\\
&x=x+z.
\end{aligned}\tag{3.5.5}$$

我们称此过程为*混合精度迭代改进*. 在用双精度计算 $r$ 时必须用原始的 $\boldsymbol{A}$. 关于 (3.5.5) 表现的基本结论总结如下.

**启示三** *如果机器精度* $\mathbf{u}$ *和条件数满足* $\mathbf{u}\approx 10^{-d}$ *和* $\kappa_\infty(\boldsymbol{A})\approx 10^q$, *则执行* (3.5.5) $k$ *次之后,* $\boldsymbol{x}$ *有大约* $\min\{d,k(d-q)\}$ *位正确的有效数字.*

精略地说, 如果 $\mathbf{u}\kappa_\infty(A)\leqslant 1$, 则迭代改进完全可给出一个全 (单) 精度正确的解. 注意到此过程是相对经济的. 每次改进的工作量为 $O(n^2)$, 相比之下原始的分解 $\boldsymbol{PA}=\boldsymbol{LU}$ 的工作量为 $O(n^3)$. 当然, 若 $\boldsymbol{A}$ 相对于机器精度是足够病态的, 则得不到任何改进.

混合精度迭代改进的一个主要缺点是它的实现是与机器相关的, 对于希望能广泛流传的软件来说, 这不会受到鼓励. 方法的另一个不足之处是需要保留 $\boldsymbol{A}$ 的原始数据.

另一方面, 混合精度迭代改进在具有累加内积 (即求 $\boldsymbol{A}$ 的行与 $\boldsymbol{x}$ 的双精度内积) 功能的机器, 常常是很容易实现的, 在机器位数不长时, 用迭代改进方法可极大地拓宽求解 $\boldsymbol{Ax}=\boldsymbol{b}$ 问题的范围.

**例 3.5.2** *如果* (3.5.5) *用于方程组*

$$\begin{bmatrix} 0.986 & 0.579 \\ 0.409 & 0.237 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} 0.235 \\ 0.107 \end{bmatrix},$$

*设* $\beta=10,t=3$. *则迭代改进给出如下解序列:*

$$\hat{\boldsymbol{x}}=\begin{bmatrix} 2.11 \\ -3.17 \end{bmatrix},\begin{bmatrix} 1.99 \\ -2.99 \end{bmatrix},\begin{bmatrix} 2.00 \\ -3.00 \end{bmatrix},\cdots$$

*精确解是* $\boldsymbol{x}=(2,-3)^{\mathrm{T}}$.

### 3.5.4 条件数估计

假设我们通过 $\boldsymbol{PA}=\boldsymbol{LU}$ 已求解 $\boldsymbol{Ax}=\boldsymbol{b}$, 现在希望确定计算解 $\hat{\boldsymbol{x}}$ 中的正确有效数字的位数. 从启示二知我们需要估计条件数 $\kappa_\infty(\boldsymbol{A})=\|\boldsymbol{A}\|_\infty\|\boldsymbol{A}^{-1}\|_\infty$. 计算 $\|\boldsymbol{A}\|_\infty$ 不成问题, 因为我们仅需用公式

$$\|\boldsymbol{A}\|_\infty=\max_{1\leqslant i\leqslant n}\sum_{j=1}^{n}|a_{ij}|.$$

难点在于因子 $\|\boldsymbol{A}^{-1}\|_\infty$. 一种想象的做法是估计 $\|\hat{\boldsymbol{X}}\|_\infty$, 其中 $\hat{\boldsymbol{X}}=(\hat{\boldsymbol{x}},\cdots,\hat{\boldsymbol{x}}_n)$ 而 $\hat{\boldsymbol{x}}_i$ 是 $\boldsymbol{Ax}_i=\boldsymbol{e}_i$ (见 3.4.9 节) 之计算解. 这一方法的不足是它的计算量: 计算 $\hat{\kappa}_\infty=\|\boldsymbol{A}\|_\infty\|\hat{\boldsymbol{X}}\|_\infty$ 的工作量大约是计算 $\hat{\boldsymbol{x}}$ 的三倍.

条件数估计的中心问题是假定已有 $\boldsymbol{PA}=\boldsymbol{LU}$ 或后面各章所给出的分解时用 $O(n^2)$ 个 flop 来估计条件数. Forsythe and Moler(SLE, 第 51 页) 所给出的一种方法是基于迭代改进以及粗略估计式 $\mathbf{u}\kappa_\infty(\boldsymbol{A})\approx\|\boldsymbol{z}\|_\infty/\|\boldsymbol{x}\|_\infty$, 其中 $\boldsymbol{z}$ 是 (3.5.5) 中对 $\boldsymbol{x}$ 的第一次修正. 虽然估计条件数的工作量为 $O(n^2)$, 但它具有迭代改进的缺点, 即机器相关.

Cline, Moler, Stewart, and Wilkinson(1979) 提出了一个没有这一缺陷的估计条件数的非常成功的方法. 它基于对以下关系式

$$\boldsymbol{Ay}=\boldsymbol{d}\Rightarrow\|\boldsymbol{A}^{-1}\|_\infty\geqslant\|\boldsymbol{y}\|_\infty/\|\boldsymbol{d}\|_\infty$$

的利用. 此方法的基本思想是选取 $\boldsymbol{d}$ 使得 $\boldsymbol{y}$ 的范数尽可能大, 然后令

$$\hat{\kappa}_\infty=\|\boldsymbol{A}\|_\infty\|\boldsymbol{y}\|_\infty/\|\boldsymbol{d}\|_\infty.$$

该方法的成功取决于 $\|\boldsymbol{y}\|_\infty/\|\boldsymbol{d}\|_\infty$ 与它的极大值 $\|\boldsymbol{A}^{-1}\|_\infty$ 的靠近程度.

考虑当 $\boldsymbol{A}=\boldsymbol{T}$ 是上三角形矩阵的情形. $\boldsymbol{d}$ 和 $\boldsymbol{y}$ 的关系完全由下面的列形式的向后消去所确定:

$$\begin{aligned}
&p(1:n)=0\\
&\textbf{for } k=n:-1:1\\
&\qquad \text{选取 } d(k)\\
&\qquad y(k)=(d(k)-p(k))/T(k,k)\\
&\qquad p(1:k-1)=p(1:k-1)+y(k)T(1:k-1,k)\\
&\textbf{end}
\end{aligned}\tag{3.5.6}$$

通常, 我们用此算法求解一给定的三角方程组 $\boldsymbol{Ty}=\boldsymbol{d}$. 但是, 这里我们自由选取右端项 $\boldsymbol{d}$, 约束条件是 $\boldsymbol{y}$ 相对于 $\boldsymbol{d}$, 尽可能大.

一种使得 $\boldsymbol{y}$ 增长的方式是从集合 $\{-1,1\}$ 中选取 $d(k)$ 使 $y(k)$ 尽可能大. 如果 $p(k)\geqslant 0$, 则取 $d(k)=-1$. 如果 $p(k)<0$, 则取 $d(k)=+1$. 换句话说, (3.5.6) 使得 $d(k)=-\mathrm{sign}(p(k))$. 由于此时 $\boldsymbol{d}$ 是形如 $d(1:n)=(\pm1,\cdots,\pm1)^{\mathrm{T}}$ 的向量, 我们得到估计式 $\hat{\kappa}_\infty=\|\boldsymbol{T}\|_\infty\|\boldsymbol{y}\|_\infty$.

如果 $d(k) \in \{-1,+1\}$ 使得同时增长 $y(k)$ 和 $p(1:k-1)+T(1:k-1,k)y(k)$，则得到一个更好的估计式. 确切地说, 在第 $k$ 步我们计算

$$
\begin{aligned}
y(k)^+ &= (1-p(k))/T(k,k) \\
s(k)^+ &= |y(k)^+| + \|p(1:k-1)+T(1:k-1,k)y(k)^+\|_1 \\
y(k)^- &= (-1-p(k))/T(k,k) \\
s(k)^- &= |y(k)^-| + \|p(1:k-1)+T(1:k-1,k)y(k)^-\|_1
\end{aligned}
$$

令

$$
y(k) = \begin{cases} y(k)^+, & \text{如果 } s(k)^+ \geqslant s(k)^- \\ y(k)^-, & \text{如果 } s(k)^+ < s(k)^-, \end{cases}
$$

这就给出了下列算法.

**算法 3.5.1 (条件数估计)** 设 $T \in \mathbb{R}^{n\times n}$ 是非奇异上三角形矩阵. 本算法计算单位 $\infty$ 范数向量 $\boldsymbol{y}$ 和标量 $\kappa$ 使得 $\|\boldsymbol{Ty}\|_\infty \approx 1/\|\boldsymbol{T}^{-1}\|_\infty$ 和 $\kappa \approx \kappa_\infty(T)$.

$p(1:n)=0$
**for** $k=n:-1:1$
  $y(k)^+ = (1-p(k))/T(k,k)$
  $y(k)^- = (-1-p(k))/T(k,k)$
  $p(k)^+ = p(1:k-1)+T(1:k-1,k)y(k)^+$
  $p(k)^- = p(1:k-1)+T(1:k-1,k)y(k)^-$
  **if** $|y(k)^+| + \|p(k)^+\|_1 \geqslant |y(k)^-| + \|p(k)^-\|_1$
    $y(k) = y(k)^+$
    $p(1:k-1) = p(k)^+$
  **else**
    $y(k) = y(k)^-$
    $p(1:k-1) = p(k)^-$
  **end**
**end**
$\kappa = \|y\|_\infty \|T\|_\infty$
$y = y/\|y\|_\infty$

此算法需要普通向后消去法几倍的工作量.

现在我们描述估计非奇异方阵 $\boldsymbol{A}$ 的条件数, 假定它的分解 $\boldsymbol{PA}=\boldsymbol{LU}$ 已知:

- 应用下三角形式的算法 3.5.1 于 $\boldsymbol{U}^{\mathrm{T}}$ 得到 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{y}=\boldsymbol{d}$ 的大范数解.
- 解三角方程组 $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{r}=\boldsymbol{y}, \boldsymbol{Lw}=\boldsymbol{Pr}, \boldsymbol{Uz}=\boldsymbol{w}$.
- $\hat{\kappa}_\infty = \|\boldsymbol{A}\|_\infty \|\boldsymbol{z}\|_\infty / \|\boldsymbol{r}\|_\infty$.

注意到 $\|\boldsymbol{z}\|_\infty \leqslant \|\boldsymbol{A}^{-1}\|_\infty \|\boldsymbol{r}\|_\infty$. 此方法基于几个直观结果, 首先, 如果 $\boldsymbol{A}$ 是病态的且 $\boldsymbol{PA}=\boldsymbol{LU}$, 则所对应的 $\boldsymbol{U}$ 通常也是病态的. 下三角形矩阵 $\boldsymbol{L}$ 一般是良态的. 于是将条件数预估方法用于 $\boldsymbol{U}$ 比用于 $\boldsymbol{L}$ 更有益. 因为向量 $\boldsymbol{r}$ 是 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}^{\mathrm{T}}\boldsymbol{r}=\boldsymbol{d}$ 之解,

所以 $\boldsymbol{r}$ 一般靠近 $\sigma_{\min}(\boldsymbol{A})$ 所对应的左奇异向量. 具有这种性质的右端项使得方程组 $\boldsymbol{A}\boldsymbol{z}=\boldsymbol{r}$ 有很大的解.

在实际中发现, 我们所简叙的条件数估计方法能给出真实的条件数比较好的量级之估计.

## 习　题

**3.5.1** 举例说明可能有多于一种的平衡矩阵的方法.

**3.5.2** 用 $\beta=10, t=2$ 运算, 和带列选主的高斯消去法求解

$$\begin{bmatrix} 11 & 15 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} 7 \\ 3 \end{bmatrix}.$$

进行一步迭代改进, 用 $t=4$ 运算计算余量 (不要忘记将计算的余量舍入到两位数).

**3.5.3** 设 $\boldsymbol{P}(\boldsymbol{A}+\boldsymbol{E})=\hat{\boldsymbol{L}}\hat{\boldsymbol{U}}$, 其中 $P$ 是置换矩阵, $\hat{\boldsymbol{L}}$ 是满足 $|\hat{l}_{ij}|\leqslant 1$ 的单位下三角形矩阵, $\hat{\boldsymbol{U}}$ 是上三角形矩阵. 证明 $\hat{\kappa}_{\infty}(\boldsymbol{A})\geqslant\|\boldsymbol{A}\|_{\infty}/(\|\boldsymbol{E}\|_{\infty}+\mu)$, 其中 $\mu=\min|\hat{u}_{ii}|$. 结论是当列选主的高斯消去法用于 $\boldsymbol{A}$ 时, 如果有小主元, 则 $\boldsymbol{A}$ 是病态的. 反之不然. (令 $\boldsymbol{A}=\boldsymbol{B}_n$.)

**3.5.4** (Kahan, 1966) 方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$, 有解 $\boldsymbol{x}=(10^{-10},-1,A)^{\mathrm{T}}$, 其中

$$\boldsymbol{A}=\begin{bmatrix} 2 & -1 & 1 \\ -1 & 10^{-10} & 10^{-10} \\ 1 & 10^{-10} & 10^{-10} \end{bmatrix},\quad \boldsymbol{b}=\begin{bmatrix} 2(1+10^{-10}) \\ -10^{-10} \\ 10^{-10} \end{bmatrix}.$$

(a) 证明如果 $(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{y}=\boldsymbol{b}$ 和 $|\boldsymbol{E}|\leqslant 10^{-8}|\boldsymbol{A}|$, 则 $|\boldsymbol{x}-\boldsymbol{y}|\leqslant 10^{-7}|\boldsymbol{x}|$. 这就是说, $\boldsymbol{A}$ 的元素中小的变化不会导致 $\boldsymbol{x}$ 的大的变化, 尽管 $\kappa_{\infty}(\boldsymbol{A})=10^{10}$. (b) 定义 $\boldsymbol{D}=\mathrm{diag}(10^{-5},10^{5},10^{5})$. 证明 $\kappa_{\infty}(\boldsymbol{DAD})\leqslant 5$. (c) 利用定理 2.7.3 来解释所发生的情况.

**3.5.5** 考虑矩阵

$$\boldsymbol{T}=\begin{bmatrix} 1 & 0 & M & -M \\ 0 & 1 & -M & M \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\quad M\in\mathbb{R}.$$

当应用 (3.5.6) 且 $d(k)=-\mathrm{sign}(\boldsymbol{P}(k))$ 时可得到什么样的 $\kappa_{\infty}(\boldsymbol{T})$ 之估计? 算法 3.5.1 得到什么样的估计? 真的 $\kappa_{\infty}(\boldsymbol{T})$ 是什么?

**3.5.6** 当算法 3.5.1 应用于 (2.7.9) 中的 $\boldsymbol{B}_n$ 时给出什么条件数估计?

## 本节注释与参考文献

以下文献是关于 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 的加权:

F. L. Bauer (1963). "Optimally Scaled Matrices," *Numer. Math.* 5, 73–87.

P. A. Businger (1968). "Matrices Which Can be Optimally Scaled," *Numer. Math.* 12, 346–348.

A. van der Sluis (1969). "Condition Numbers and Equilibration Matrices," *Numer. Math.* 14, 14–23.

A. van der Sluis (1970). "Condition, Equilibration, and Pivoting in Linear Algebraic Systems," *Numer. Math.* 15, 74–86.

C. McCarthy and G. Strang (1973). "Optimal Conditioning of Matrices," *SIAM J. Num. Anal.* 10, 370–388.

T. Fenner and G. Loizou (1974). "Some New Bounds on the Condition Numbers of Optimally Scaled Matrices," *J. ACM* 21, 514–524.

G. H. Golub and J. M. Varah (1974). "On a Characterization of the Best $L_2$-Scaling of a Matrix," *SIAM J. Num. Anal.* 11, 472–479.

R. Skeel (1979). "Scaling for Numerical Stability in Gaussian Elimination," *J. ACM.* 26, 494–526.

R. Skeel (1981). "Effect of Equilibration on Residual Size for Partial Pivoting," *SIAM J. Num. Anal.* 18, 449–455.

加权的部分困难是关于选择什么样的范数来估计误差. 这一经常被忽视的有趣问题之讨论可见:

W. Kahan(1966). "Numerical Linear Algebra," *Canadian Math. Bull.* 9, 757–801.

关于迭代改进以及相关问题的严格分析可见:

M. Jankowski and M. Wozniakowski (1977). "Iterative Refinement Implies Numerical Stability," *BIT* 17, 303–311.

C. B. Moler (1967). "Iterative Refinement in Floating Point," *J. ACM* 14, 316–371.

R. D. Skeel (1980). "Iterative Refinement Implies Numerical Stability for Gaussian Elimination," *Math. Comp.* 35, 817–832.

G. W. Stewart (1981). "On the Implicit Deflation of Nearly Singular Systems of Linear Equations," *SIAM J. Sci.* and *Stat. Comp.* 2, 136–140.

我们叙述的条件数估计方法由下文给出:

A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson (1979). "An Estimate for the Condition Number of a Matrix," *SIAM J. Num. Anal.* 16, 368–375.

关于条件数估计的其他文章包括:

C. G. Broyden (1973). "Some Condition Number Bounds for the Gaussian Elimination Process," *J. Inst. Math. Applic.* 12, 273–286.

F. Lemeire (1973). "Bounds for Condition Numbers of Triangular Value of a Matrix," *Lin. Alg. and Its Applic.* 11, 1–2.

R. S. Varga (1976). "On Diagonal Dominance Arguments for Bounding $\|A^{-1}\|_\infty$," *Lin. Alg. and Its Applic.* 14, 211–217.

G. W. Stewart (1980). "The Efficient Generation of Random Orthogonal Matrices with an Application to Condition Estimators," *SIAM J. Num. Anal.* 17, 403–409.

D. P. O'Larey (1980). "Estimating Matrix Condition Numbers," *SIAM J. Sci. Stat. Comp.* 1, 205–209.

R. G. Grimes and J. G. Lewis (1981). "Condition Number Estimation for Sparse Matrices," *SIAM J. Sci. and Stat. Comp.* 2, 384–388.

A. K. Cline, A. R. Conn, and C. Van Loan (1982). "Generalizing the LINPACK Condition Estimator," in *Numerical Analysis*, ed., J. P. Hennart, Lecture Notes in Mathematics no. 909, Springer-Verlag, New York.

A. K. Cline and R. K. Rew (1983). "A set of Counter examples to Three Condition Number Estimators," *SIAM J. Sci and Stat. Comp.* 4, 602–611.

W. Hager (1984). "Condition Estimates," *SIAM J. Sci. and Stat. Comp.* 5, 311–316.

N. J. Higham (1987). "A Survey of Condition Number Estimation for Triangular Matrices," *SIAM Review* 29, 575–596.

N. J. Higham (1988). "Fortran Codes for Estimating the One-norm of a Real or Complex Matrix. with Applications to Condition Estimation," *ACM Trans. Math. Soft.* 14, 381–396.

N. J. Higham (1988). "FORTRAN Codes for Estimating the One-Norm of a Real or Complex Matrix with Applications to Condition Estimation (Algorithm 674)," *ACM Trans. Math. Soft.* 14, 381–396.

C. H. Bischof (1990). "Incremental Condition Estimation," *SIAM J. Matrix Anal. Appl.* 11, 644–659.

C. H. Bischof (1990). "Incremental Condition Estimation for Sparse Matrices," *SIAM J. Matrix Anal. Appl.* 11, 312–322.

G. Auchmuty (1991). "A Posteriori Error Estimates for Linear Equation," *Numer. Math.* 61, 1–6.

N. J. Higham (1993). "Optimization by Direct Search in Matrix Computations," *SIAM J. Matrix Anal. Appl.* 14, 317–333.

D. J. Higham (1995). "Condition Numbers and Their Condition Numbers," *Lin. Alg. and Its Applic.* 214, 193–213.

# 第 4 章　特殊线性方程组

数值分析的一个基本原则是：求解任一问题都应利用它的结构特性. 在数值线性代数中, 这意味着当问题中出现诸如对称性、正定性和稀疏性等特性时, 需要将适用于求解一般矩阵的算法修改, 使其效率更高. 这是本章的主题, 我们的主要目的是设计一些计算特殊 LU 分解的专用算法.

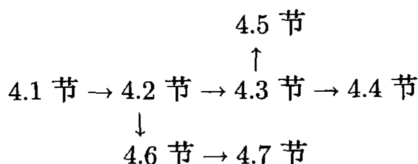
首先我们指出当 $\boldsymbol{A}$ 对称时三角形因子 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 的关系, 这是在 4.1 节中通过分析 $\mathrm{LDM^T}$ 分解得到的. 随后在 4.2 节中我们将注意力转到 $\boldsymbol{A}$ 为对称正定这一重要的情形, 导出稳定 Cholesky 分解. 这一节也讨论了非对称正定方程组. 4.3 节讨论了高斯消去法和其他一些分解法的带状形式. 之后我们讨论了 $\boldsymbol{A}$ 对称但不定这一有趣情形. 4.4 节中我们对此问题的处理显示了数值分析学者对选主元既爱又恨. 我们喜欢选主元因为它能保证稳定性, 但由于它会破坏问题的结构而讨厌它. 幸运的是关于对称非定方程组此冲突已有一个圆满的解决办法.

任何分块带状矩阵, 其本身也是带状矩阵, 故 4.3 节的方法仍适用. 但是坚持这样的观点有时并不值得. 在 4.5 节中我们以分块三对角方程组为例来说明这种情况, 同时还讨论了一些其他的分块方程组.

在最后两节, 将研究一些很有意义的 $O(n^2)$ 的算法, 这些算法可用来解 Vandermonde 方程组和 Toeplitz 方程组.

**预备知识**

阅读本章需要掌握第 1 章、2.1 节 ~2.5 节、2.7 节和第 3 章的知识, 本章各节间的关系如下：




补充参考文献包括 George and Liu (1981), Gill, Murry, and Wright (1991), Higham (1996), Trefethen and Bau (1996), Demmel (1996). 本章用到的 MATLAB 函数有：chol,tril,triu,vander,toeplitz,fft. 与 LAPACK 相关的例程包括：

| LAPACK：一般带状矩阵 | |
|---|---|
| _GBSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _CGBCON | 求条件数 |
| _GBRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ 并给出误差界 |
| _GBSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ 并给出条件数 |
| _GBTRF | $\boldsymbol{PA}=\boldsymbol{LU}$ |
| _GBTRS | 用 $\boldsymbol{PA}=\boldsymbol{LU}$ 解 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ |
| _GBEQU | 平衡问题 |

| | LAPACK：一般三对角矩阵 |
|---|---|
| _GTSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _GTCON | 求条件数 |
| _GTRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ 并给出误差界 |
| _GTSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ 并给出条件数 |
| _GTTRF | $\boldsymbol{PA}=\boldsymbol{LU}$ |
| _GTTRS | 用 $\boldsymbol{PA}=\boldsymbol{LU}$ 解 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{H}}\boldsymbol{X}=\boldsymbol{B}$ |

| | LAPACK：满对称正定矩阵 |
|---|---|
| _POSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _POCON | 用 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 求条件数 |
| _PORFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出误差界 |
| _POSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出条件数 |
| _POTRF | $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ |
| _POTRS | 用 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _POTRI | $\boldsymbol{A}^{-1}$ |
| _POEQU | 平衡问题 |

| | LAPACK：带状对称正定矩阵 |
|---|---|
| _PBSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _PBCON | 用 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 求条件数 |
| _PBRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出误差界 |
| _PBSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出条件数 |
| _PBTRF | $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ |
| _PBTRS | 用 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 解 $\boldsymbol{AX}=\boldsymbol{B}$ |

| | LAPACK：三对角对称正定矩阵 |
|---|---|
| _PTSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _PTCON | 用 $\boldsymbol{A}=\boldsymbol{LDL}^{\mathrm{T}}$ 求条件数 |
| _PTRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出误差界 |
| _PTSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出条件数 |
| _PTTRF | $\boldsymbol{A}=\boldsymbol{LDL}^{\mathrm{T}}$ |
| _PTTRS | 利用 $\boldsymbol{A}=\boldsymbol{LDL}^{\mathrm{T}}$ 解 $\boldsymbol{AX}=\boldsymbol{B}$ |

| | LAPACK：满对称非定矩阵 |
|---|---|
| _SYSV | 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _SYCON | 用 $\boldsymbol{PAP}^{\mathrm{T}}=\boldsymbol{LDL}^{\mathrm{T}}$ 求条件数 |
| _SYRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出误差界 |
| _SYSVX | 解 $\boldsymbol{AX}=\boldsymbol{B}$ 并给出条件数 |
| _SYTRF | $\boldsymbol{PAP}^{\mathrm{T}}=\boldsymbol{LDL}^{\mathrm{T}}$ |
| _SYTRS | 利用 $\boldsymbol{PAP}^{\mathrm{T}}=\boldsymbol{LDL}^{\mathrm{T}}$ 解 $\boldsymbol{AX}=\boldsymbol{B}$ |
| _SYTRI | $\boldsymbol{A}^{-1}$ |

| | LAPACK：三角形带状矩阵 |
|---|---|
| _TBCON | 用 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 求条件数 |
| _TBRFS | 改进 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}$ 并给出误差界 |
| _TBTRS | 解 $\boldsymbol{AX}=\boldsymbol{B}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{B}$ |

# 4.1 LDM$^{\mathrm{T}}$ 和 LDL$^{\mathrm{T}}$ 分解

我们需要发展一种能利用结构的求解对称问题 $\boldsymbol{AX}=\boldsymbol{b}$ 的方法. 为此, 我们建立一种变形的 LU 分解, 将 $\boldsymbol{A}$ 分解为三个因子的乘积 $\boldsymbol{LDM}^{\mathrm{T}}$, 其中 $\boldsymbol{D}$ 是对角矩阵, $\boldsymbol{L}$ 和 $\boldsymbol{M}$ 是单位下三角形矩阵. 一旦得到此分解, 那么就可以通过 $O(n^2)$ 次浮点运算求解 $\boldsymbol{Ly}=\boldsymbol{b}$ (向前消去), $\boldsymbol{Dz}=\boldsymbol{y}$ 和 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{z}$ (向后迭代) 来得到 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解. 引入 LDM$^{\mathrm{T}}$ 分解的目的是为 $\boldsymbol{A}$ 是对称阵的情况做铺垫, 此时 $\boldsymbol{A}=\boldsymbol{A}^{\mathrm{T}}$, 则 $\boldsymbol{L}=\boldsymbol{M}$. 基于这种分解方法的工作量仅为高斯消去法的一半. 后继的内容讨论选主元问题.

### 4.1.1 LDM$^{\mathrm{T}}$ 分解

首先看一下 LDM$^{\mathrm{T}}$ 分解与 LU 分解之间的联系.

**定理 4.1.1** *如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 的所有顺序主子矩阵都是非奇异的, 则存在唯一的单位下三角形矩阵 $\boldsymbol{L}$ 和 $\boldsymbol{M}$ 以及唯一的对角矩阵 $\boldsymbol{D}=\mathrm{diag}(d_1,\cdots,d_n)$ 满足 $\boldsymbol{A}=\boldsymbol{LDM}^{\mathrm{T}}$.*

**证明** 由定理 3.2.1 知, 存在 $A$ 的分解 $\boldsymbol{A}=\boldsymbol{LU}$. 令 $\boldsymbol{D}=\mathrm{diag}(d_1,\cdots,d_n)$, 其中对所有 $i=1:n$, 有 $d_i=u_{ii}$. 注意到 $\boldsymbol{D}$ 是非奇异的, $\boldsymbol{M}^{\mathrm{T}}=\boldsymbol{D}^{-1}\boldsymbol{U}$ 是单位上三角形矩阵. 因此 $\boldsymbol{A}=\boldsymbol{LU}=\boldsymbol{LD}(\boldsymbol{D}^{-1}\boldsymbol{U})=\boldsymbol{LDM}^{\mathrm{T}}$. 唯一性可由定理 3.2.1 中 LU 分解的唯一性得出. □

由上面的证明过程可以看出, LDM$^{\mathrm{T}}$ 分解可由如下方式得到: 首先通过高斯消去法计算 $\boldsymbol{A}=\boldsymbol{LU}$, 然后由 $\boldsymbol{U}=\boldsymbol{DM}^{\mathrm{T}}$ 来解出 $\boldsymbol{D}$ 和 $\boldsymbol{M}$. 然而, 通过直接计算 $\boldsymbol{L},\boldsymbol{D}$ 和 $\boldsymbol{M}$ 能导出另一个有趣的算法.

假设对某满足 $1\leqslant j\leqslant n$ 的 $j$, 已知 $\boldsymbol{L}$ 的前 $j-1$ 列、$\boldsymbol{D}$ 的对角元 $d_1,d_2\cdots d_{j-1}$ 和 $\boldsymbol{M}$ 的前 $j-1$ 行. 为求 $L(j+1:n,j),M(j,1:j-1)$ 和 $d_j$, 我们取出方程 $\boldsymbol{A}=\boldsymbol{LDM}^{\mathrm{T}}$ 中第 $j$ 列的等式, 具体地说,

$$A(1:n,j)=\boldsymbol{Lv},\tag{4.1.1}$$

其中 $\boldsymbol{v}=\boldsymbol{DM}^{\mathrm{T}}\boldsymbol{e}_j$. (4.1.1) 式的向量的 "上半部" 将 $v(1:j)$ 定义为已知的下三角方程组:

$$L(1:j,1:j)v(1:j)=A(1:j,j)$$

的解. 一旦求得 $\boldsymbol{v}$, 则可以计算得

$$d(j)=v(j)$$

$$M(j,i)=v(i)/d(i),\quad i=1:j-1.$$

(4.1.1) 式的 "下半部" 有关系式

$$L(j+1:n,1:j)v(1:j)=A(j+1:n,j),$$

它重新组织后可用来求 $\boldsymbol{L}$ 的第 $j$ 列:

$$L(j+1:n,j)v(j) = A(j+1:n,j) - L(j+1:n,1:j-1)v(1:j-1).$$

因此, $L(j+1:n,1:j)$ 是加权 gaxpy 运算. 综上我们得到

$$
\begin{array}{l}
\textbf{for } j=1:n \\
\quad \text{从 } L(1:j,1:j)v(1:j) = A(1:j,j) \text{ 解出 } v(1:j) \\
\quad \textbf{for } i=1:j-1 \\
\quad\quad M(j,i) = v(i)/d(i) \\
\quad \textbf{end} \\
\quad d(j) = v(j) \\
\quad L(j+1:n,j) = (A(j+1:n,j) - L(j+1:n,1:j-1)v(1:j-1))/v(j) \\
\textbf{end}
\end{array}
\tag{4.1.2}
$$

和 LU 分解类似, $\boldsymbol{A}$ 可用因子 $\boldsymbol{L}, \boldsymbol{D}, \boldsymbol{M}$ 覆盖. 如果采用列形式的向前消去法来解 $v(1:j)$, 则可得到以下算法.

**算法 4.1.1 ($\mathbf{LDM^T}$)** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 存在 LU 分解, 则本算法计算满足 $\boldsymbol{A} = \boldsymbol{LDM}^{\mathrm{T}}$ 的单位下三角形矩阵 $\boldsymbol{L}, \boldsymbol{M}$ 和对角矩阵 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n)$, $\boldsymbol{A}$ 中元素 $a_{ij}$ 分别由 $l_{ij}(i>j), d_i(i=j), m_{ji}(i<j)$ 所覆盖.

$$
\begin{array}{l}
\textbf{for } j=1:n \\
\quad \{\text{解 } L(1:j,1:j)v(1:j) = A(1:j,j).\} \\
\quad v(1:j) = A(1:j,j) \\
\quad \textbf{for } k=1:j-1 \\
\quad\quad v(k+1:j) = v(k+1:j) - v(k)A(k+1:j,k) \\
\quad \textbf{end} \\
\quad \{\text{计算 } M(j,1:j-1) \text{ 且存于 } A(1:j-1,j).\} \\
\quad \textbf{for } i=1:j-1 \\
\quad\quad A(i,j) = v(i)/A(i,i) \\
\quad \textbf{end} \\
\quad \{\text{存 } d(j) \text{ 于 } A(j,j).\} \\
\quad A(j,j) = v(j) \\
\quad \{\text{计算 } L(j+1:n,j) \text{ 且存于 } A(j+1:n,j).\} \\
\quad \textbf{for } k=1:j-1 \\
\quad\quad A(j+1:n) = A(j+1:n,j) - v(k)A(j+1:n,k) \\
\quad \textbf{end} \\
\quad A(j+1:n,j) = A(j+1:n,j)/v(j) \\
\textbf{end}
\end{array}
$$

本算法的工作量与 LU 分解相同, 约为 $2n^2/3$ 个 flop.

可以证明, 利用算法 4.1.1 和 3.1 节中的一般三角方程组解法所得到的 $\boldsymbol{Ax}=\boldsymbol{b}$ 之计算解 $\hat{\boldsymbol{x}}$ 满足扰动方程组 $(\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{x}}=\boldsymbol{b}$, 其中

$$|\mathbf{E}| \leqslant n\mathbf{u}(3|\boldsymbol{A}|+5|\hat{\boldsymbol{L}}||\hat{\boldsymbol{D}}||\hat{\boldsymbol{M}}^{\mathrm{T}}|)+O(\mathbf{u}^2), \tag{4.1.3}$$

$\hat{\boldsymbol{L}},\hat{\boldsymbol{D}},\hat{\boldsymbol{M}}$ 分别是 $\boldsymbol{L},\boldsymbol{D},\boldsymbol{M}$ 的计算解.

如前一章考虑的 LU 分解一样, 除非进行选主元处理, 否则 (4.1.3) 式的上界将没有限制. 因此, 为使算法 4.1.1 成为实际可行的算法, 应改为计算形如 $\boldsymbol{PA}=\boldsymbol{LDM}^{\mathrm{T}}$ 的分解, 其中 $\boldsymbol{P}$ 是能够使 $\boldsymbol{L}$ 满足 $|l_{ij}|\leqslant 1$ 的置换矩阵. 由于解决这个问题非常简单, 且我们引入 $\mathrm{LDM}^{\mathrm{T}}$ 分解的目的是为寻求求解对称方程组的特殊方法, 因此, 我们将不追求其细节.

**例 4.1.1**

$$\boldsymbol{A}=\begin{bmatrix}10&10&20\\20&25&40\\30&50&61\end{bmatrix}=\begin{bmatrix}1&0&0\\2&1&0\\3&4&1\end{bmatrix}\begin{bmatrix}10&0&0\\0&5&0\\0&0&1\end{bmatrix}\begin{bmatrix}1&1&2\\0&1&0\\0&0&1\end{bmatrix},$$

按算法 4.1.1 执行完后, $A$ 被覆盖为

$$\boldsymbol{A}=\begin{bmatrix}10&1&2\\2&5&0\\3&4&1\end{bmatrix}.$$

### 4.1.2 对称性和 $\mathrm{LDL}^{\mathrm{T}}$ 分解

如果 $\boldsymbol{A}$ 是对称的, 则 $\mathrm{LDM}^{\mathrm{T}}$ 分解中有些是多余的.

**定理 4.1.2** *如果 $\boldsymbol{A}=\boldsymbol{LDM}^{\mathrm{T}}$ 是非奇异对称矩阵 $\boldsymbol{A}$ 的 $\mathrm{LDM}^{\mathrm{T}}$ 分解, 则 $\boldsymbol{L}=\boldsymbol{M}$.*

**证明**　矩阵 $\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{M}^{-\mathrm{T}}=\boldsymbol{M}^{-1}\boldsymbol{L}\boldsymbol{D}$ 对称且是下三角的, 因此它是对角矩阵. 因为 $\boldsymbol{D}$ 是非奇异的, 故 $\boldsymbol{M}^{-1}\boldsymbol{L}$ 也是对角矩阵. 而 $\boldsymbol{M}^{-1}\boldsymbol{L}$ 是单位下三角形矩阵, 因此 $\boldsymbol{M}^{-1}\boldsymbol{L}=\boldsymbol{I}$. □

从以上结论可知, 算法 4.1.1 应用于对称矩阵时, 工作量可以减半. 在第 $j$ 步, 由于 $\boldsymbol{M}=\boldsymbol{L}$ 且假定 $\boldsymbol{L}$ 的前 $j-1$ 列已知, 则 $M(j,1:j-1)$ 也为已知. 回想 (4.1.2) 中的第 $j$ 步, 向量 $v(1:j)$ 是由 $\boldsymbol{DM}^{\mathrm{T}}e_j$ 的前 $j$ 个分量定义的. 由于 $\boldsymbol{M}=\boldsymbol{L}$, 故

$$v(1:j)=\begin{bmatrix}d(1)L(j,1)\\ \vdots\\ d(j-1)L(j,j-1)\\ d(j)\end{bmatrix}.$$

于是, 向量 $v(1:j-1)$ 可通过对 $\boldsymbol{L}$ 的第 $j$ 行做简单加权来得到, 由 $L(1:j,1:j)v=A(1:j,j)$ 的第 $j$ 个方程, 有关系式 $v(j)=A(j,j)-L(j,1:j-1)v(1:j-1)$, 故有:

```
for j = 1 : n
    for i = 1 : j - 1
        v(i) = L(j, i)d(i)
    end
    v(j) = A(j, j) - L(j, 1 : j - 1)v(1 : j - 1)
    d(j) = v(j)
    L(j + 1 : n, j) = (A(j + 1 : n, j) - L(j + 1 : n, 1 : j - 1)v(1 : j - 1))/v(j)
end
```

重新整理上面所述得下列算法.

**算法 4.1.2 (LDL$^{\mathrm{T}}$)** 如果 $A \in \mathbb{R}^{n\times n}$ 对称且存在 LU 分解, 则本算法计算满足 $A = LDL^{\mathrm{T}}$ 的单位下三角形矩阵 $L$ 和对角矩阵 $D = \mathrm{diag}(d_1, \cdots, d_n)$, $A$ 的元素 $a_{ij}$ 由 $l_{ij}(j > j)$ 和 $d_i(i = j)$ 所覆盖.

```
for j = 1 : n
    {计算 v(1 : j).}
    for i = 1 : j - 1
        v(i) = A(j, i)A(i, i)
    end
    v(j) = A(j, j) - A(j, 1 : j - 1)v(1 : j - 1)
    {储存 d(j) 且计算 L(j + 1 : n, j).}
    A(j, j) = v(j)
    A(j + 1 : n, j) = (A(j + 1 : n, j) - A(j + 1 : n, 1 : j - 1)v(1 : j - 1))/v(j)
end
```

本算法需 $n^3/3$ 个 flop, 大约是高斯消去法的一半.

在下一节, 我们证明, 当 $A$ 对称且正定时, 算法 4.1.2 不但能够顺利执行完, 而且十分稳定. 如果 $A$ 对称但非正定, 则需要选主元, 4.4 节给出了相关的方法.

**例 4.1.2**

$$A = \begin{bmatrix} 10 & 20 & 30 \\ 20 & 45 & 80 \\ 30 & 80 & 171 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{bmatrix} \begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$

按算法 4.1.2 执行完后, $A$ 被覆盖为

$$A = \begin{bmatrix} 10 & 20 & 30 \\ 2 & 5 & 80 \\ 3 & 4 & 1 \end{bmatrix}$$

## 习 题

**4.1.1** 证明非奇异矩阵 $A$ 的 LDM$^{\mathrm{T}}$ 分解如果存在则必唯一.

**4.1.2** 修改算法 4.1.1 使其计算形如 $PA = LDM^{\mathrm{T}}$ 的分解, 其中 $L$ 和 $M$ 均为单位下三角形矩阵. $D$ 是对角矩阵, $P$ 为置换矩阵且满足 $|l_{ij}| \leqslant 1$.

**4.1.3** 假定 $n \times n$ 的对称矩阵 $A$ 按如下方式存储于向量 $c$ 中: $c = (a_{11}, a_{21}, \cdots, a_{n1}, a_{22}, \cdots, a_{n2}, \cdots, a_{nn})$. 按 $A$ 的这种存储结构重写算法 4.1.2, 尽可能地把下标换算移到内循环外.

**4.1.4** 将 $A$ 按对角线存储重写算法 4.1.2. 参见 1.2.8 节.

### 本节注释与参考文献

在避免外积修正方面, 算法 4.1.1 与 Crout 和 Doolittle 的方法是相关的. 参阅 Fox (1964) 的第 4 章或 Stewart (1973, 131–149). 一个 Agol 算法请参阅:

H. J. Bowdler, R. S. Martin, G. Peters, and J. H. Wilkinson (1966), "Solution of Real and Complex Systems of Linear Equations," *Numer. Math.* 8, 217–234.

和

G. E. Forsythe (1960). "Crout with Pivoting," *Comm. ACM* 3, 507–508.

W. M. McKeeman (1962). "Crout with Equilibration and Iteration," *Comm. ACM* 5, 553–555.

如同算法一样, 误差分析和扰动理论也可以利用结构特性, 见:

M. Arioli, J. Demmel, and I. Duff(1989). "Solving Sparse Linear Systems with Sparse Backward Error," *SIAM J. Matrix Anal. Appl.* 10, 165–190.

J. R. Bunch, J. W. Demmel, and C. F. Van Loan (1989). "The Strong Stability of Algorithms for Solving Symmetric Linear Systems," *SIAM J. Matrix Anal. Appl.* 10, 494–499.

A. Barrlund (1991), "Perturbation Bounds for the $LDL^{\mathrm{T}}$ and $LU$ Decompositions," *BIT* 31, 358–363.

D. J. Higham and N. J. Higham (1992). "Backward Error and Condition of Structured Linear Systems," *SIAM J. Matrix Anal. Appl.* 13, 162–175.

## 4.2 正定方程组

如果对所有非零向量 $x \in \mathbb{R}^n$ 都有 $x^{\mathrm{T}}Ax > 0$, 则称矩阵 $A \in \mathbb{R}^{n\times n}$ 是正定的. 正定方程组是特殊 $Ax = b$ 问题中最重要的一类. 考虑 $2 \times 2$ 对称矩阵的情形, 如

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

是正定的, 则

$$\begin{aligned}
x = (1,0)^{\mathrm{T}} &\Rightarrow x^{\mathrm{T}}Ax = a_{11} > 0, \\
x = (0,1)^{\mathrm{T}} &\Rightarrow x^{\mathrm{T}}Ax = a_{22} > 0, \\
x = (1,1)^{\mathrm{T}} &\Rightarrow x^{\mathrm{T}}Ax = a_{11} + 2a_{12} + a_{22} > 0, \\
x = (1,-1)^{\mathrm{T}} &\Rightarrow x^{\mathrm{T}}Ax = a_{11} - 2a_{12} + a_{22} > 0.
\end{aligned}$$

由后两个方程推知 $|a_{12}| \leqslant (a_{11}+a_{22})/2$. 由这些结果可知 $\boldsymbol{A}$ 中最大元素位于对角线上且为正. 此结论是普遍成立的. 一个对称正定矩阵有一条"重"对角线, 尽管这样的矩阵不如对角占优矩阵那样明显地将重量集中在对角线上, 但在计算中同样可以省略掉选主元的过程, 在这点上二者是等效的.

我们首先给出正定矩阵的一些性质, 再讨论在非对称情形时它对选主元的影响. 然后集中精力设计有效的 Cholesky 分解使其稳定地对一个对称正定矩阵 $\boldsymbol{A}$ 进行分解. 给出的算法包括 gaxpy, 外积和分块三种形式. 最后简单讨论半正定矩阵的情形.

### 4.2.1 正定性

假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是正定的. 显然一个正定矩阵是非奇异的, 否则可以找到一个非零向量 $\boldsymbol{x}$, 使 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}=0$. 由二次型 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$ 的非负性可推出以下很多结论.

**定理 4.2.1** 如是 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是正定的, $\boldsymbol{X} \in \mathbb{R}^{n\times k}$ 的秩为 $k$, 则 $\boldsymbol{B}=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X} \in \mathbb{R}^{k\times k}$ 也是正定的.

**证明** 如果 $\boldsymbol{z} \in \mathbb{R}^k$ 满足 $0 \geqslant \boldsymbol{z}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{z}=(\boldsymbol{X}\boldsymbol{z})^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{X}\boldsymbol{z})$, 则 $\boldsymbol{X}\boldsymbol{z}=0$. 但因 $\boldsymbol{X}$ 是列满秩的, 故 $\boldsymbol{z}=0$. □

**推论 4.2.2** 如果 $\boldsymbol{A}$ 是正定的, 则其所有的主子矩阵均为正定的. 特别地, 所有的对角元均大于零.

**证明** 如果 $\boldsymbol{v} \in \mathbb{R}^k$ 是整数向量, 且 $1 \leqslant v_1 < \cdots < v_k \leqslant n$, 则 $\boldsymbol{X}=I_n(:,v)$ 是由单位矩阵的第 $v_1 \cdots v_k$ 列组成的秩为 $k$ 的矩阵. 由定理 4.2.1 知 $A(v,v)=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 是正定的. □

**推论 4.2.3** 如果 $\boldsymbol{A}$ 是正定的, 则 $\boldsymbol{A}$ 的分解 $\boldsymbol{A}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{M}^{\mathrm{T}}$ 存在, 且 $\boldsymbol{D}=\mathrm{diag}(d_1,\cdots,d_n)$ 的对角元均大于零.

**证明** 由推论 4.2.2 知子矩阵 $A(1:k,1:k)$ 对于 $k=1:n$ 是非奇异的, 因此由定理 4.1.1 可知存在分解 $\boldsymbol{A}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{M}^{\mathrm{T}}$. 如果在定理 4.2.1 中令 $\boldsymbol{X}=\boldsymbol{L}^{-\mathrm{T}}$, 则 $\boldsymbol{B}=\boldsymbol{D}\boldsymbol{M}^{\mathrm{T}}\boldsymbol{L}^{-\mathrm{T}}=\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}}$ 是正定的. 由于 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{L}^{-\mathrm{T}}$ 是单位上三角形矩阵, 所以 $\boldsymbol{B}$ 和 $\boldsymbol{D}$ 的对角元相同, 故对角元必大于零. □

在实际中, 有几种典型情况会产生正定矩阵.

- 二次型是由物理原理保证为正定的能量函数.
- 矩阵 $\boldsymbol{A}$ 等于一个叉积 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$, 其中 $\boldsymbol{X}$ 是列满秩的 (正定性是由定理 4.2.1 中令 $\boldsymbol{A}=\boldsymbol{I}_n$ 得出的).
- $\boldsymbol{A}$ 和 $\boldsymbol{A}^{\mathrm{T}}$ 均为对角占优的且每个 $a_{ii}$ 都大于零.

### 4.2.2 非对称正定方程组

仅仅存在 $\mathrm{LDM}^{\mathrm{T}}$ 分解还不足以意味着它的计算就是可取的, 因为分解中的因子可能会有大的不能接受的元素. 例如, 如果 $\varepsilon > 0$, 则矩阵

$$A=\begin{bmatrix}\varepsilon & m\\ -m & \varepsilon\end{bmatrix}=\begin{bmatrix}1 & 0\\ -m/\varepsilon & 1\end{bmatrix}\begin{bmatrix}\varepsilon & 0\\ 0 & \varepsilon+m^2/\varepsilon\end{bmatrix}\begin{bmatrix}1 & m/\varepsilon\\ 0 & 1\end{bmatrix}$$

是正定的. 但如果 $m/\varepsilon \gg 1$, 那么最好进行选主元.

下面的结果指出了计算正定矩阵的 $\mathrm{LDM}^{\mathrm{T}}$ 分解中何时元素会增大.

**定理 4.2.4** 假设 $A\in\mathbb{R}^{n\times n}$ 是正定的, 令 $T=(A+A^{\mathrm{T}})/2$, $S=(A-A^{\mathrm{T}})/2$, 如果 $A=LDM^{\mathrm{T}}$, 则

$$\||L||D||M^{\mathrm{T}}|\|_{\mathrm{F}} \leqslant n(\|T\|_2+\|ST^{-1}S\|_2). \tag{4.2.1}$$

**证明** 参见 Golub and Van Loan (1979). □

本定理指出了在什么情况下不选主元也是安全的. 假定计算得的因子 $\hat{L},\hat{D},\hat{M}$ 满足

$$\||\hat{L}||\hat{D}||\hat{M}^{\mathrm{T}}|\|_{\mathrm{F}} \leqslant c\||L||D||M^{\mathrm{T}}|\|_{\mathrm{F}}, \tag{4.2.2}$$

其中 $c$ 是大小适中的常数. 由 (4.2.1) 和 3.3 节的分析, 如利用这些因子来计算 $Ax=b$, 则计算解 $\hat{x}$ 满足 $(A+E)\hat{x}=b$, 且有

$$\|E\|_{\mathrm{F}} \leqslant \mathbf{u}(3n\|A\|_{\mathrm{F}}+5cn^2(\|T\|_2+\|ST^{-1}S\|_2))+O(\mathbf{u}^2). \tag{4.2.3}$$

容易看出 $\|T\|_2\leqslant\|A\|_2$, 于是, 只要

$$\Omega=\frac{\|ST^{-1}S\|_2}{\|A\|_2} \tag{4.2.4}$$

不太大, 则不选主元也是安全的. 换言之, 与对称部分 $T$ 的条件相比, 非对称部分 $S$ 的范数不太大时, 不选主元是可靠的. 有时在具体应用中可估计 $\Omega$. 明显的例子是当 $A$ 对称时, 有 $\Omega=0$.

### 4.2.3 对称正定方程组

将上述结果用于对称正定方程组. 我们知分解 $A=LDL^{\mathrm{T}}$ 存在并且其计算是稳定的. 但是, 此时还有另外一种分解方法.

**定理 4.2.5 (Cholesky 分解)** 如果 $A\in\mathbb{R}^{n\times n}$ 是对称正定的, 则存在唯一的一个对角元全部大于零的下三角形矩阵 $G\in\mathbb{R}^{n\times n}$, 满足 $A=GG^{\mathrm{T}}$.

**证明** 由定理 4.1.2. 存在单位下三角形矩阵 $L$ 和对角矩阵 $D=\mathrm{diag}(d_1,\cdots,d_n)$ 使得 $A=LDL^{\mathrm{T}}$. 由于 $d_k>0$ 大于零, 则矩阵 $G=L\mathrm{diag}(\sqrt{d_1},\cdots,\sqrt{d_n})$ 是对角元大于零的实下三角矩阵. 它同时满足 $A=GG^{\mathrm{T}}$. 唯一性由 $\mathrm{LDL}^{\mathrm{T}}$ 分解的唯一性可推得. □

分解 $A=GG^{\mathrm{T}}$ 被称为 Cholesky 分解, $G$ 被称为 Cholesky 三角矩阵. 如果我们计算 Cholesky 分解, 然后解三角形方程组 $Gy=b$ 和 $G^{\mathrm{T}}x=y$, 则 $b=Gy=G(G^{\mathrm{T}}x)=(GG^{\mathrm{T}})x=Ax$.

在定理 4.2.5 中, Cholesky 分解的证明是构造性的. 然而, 可以通过利用方程 $\boldsymbol{A}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 来得到计算 Cholesky 三角阵的更有效的方法. 在以下几小节中我们将说明有好几种方式来做到这一点.

**例 4.2.1** 矩阵

$$\begin{bmatrix} 2 & -2 \\ -2 & 5 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}\begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} \sqrt{2} & 0 \\ -\sqrt{2} & \sqrt{3} \end{bmatrix}\begin{bmatrix} \sqrt{2} & -\sqrt{2} \\ 0 & \sqrt{3} \end{bmatrix}$$

是正定的.

### 4.2.4 基于 Gaxpy 的 Cholesky 分解

我们首先导出一个含有大量 gaxpy 运算的 Cholesky 分解的实现方法. 比较等式 $\boldsymbol{A}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 的第 $j$ 列可得

$$A(:,j)=\sum_{k=1}^{j} G(j,k)G(:,k).$$

也就是说,

$$G(j,j)G(:,j)=A(:,j)-\sum_{k=1}^{j-1} G(j,k)G(:,k)\equiv v. \tag{4.2.5}$$

如果 $G$ 的前 $j-1$ 列已知, 则可计算出 $v$. 由 (4.2.5) 中各元素间的相等关系推出

$$G(j:n,j)=v(j:n)/\sqrt{v(j)}.$$

这是加权 gaxpy 运算, 于是得到基于 gaxpy 运算的计算 Cholesky 分解的方法:

```
for j = 1 : n
    v(j : n) = A(j : n, j)
    for k = 1 : j - 1
        v(j : n) = v(j : n) - G(j, k)G(j : n, k)
    end
    G(j : n, j) = v(j : n)/sqrt(v(j))
end
```

可以在计算过程中用 $G$ 覆盖 $A$ 的下三角部分.

**算法 4.2.1 (Gholesky 分解: 基于 gaxpy 运算)** 给出对称正定矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$, 本算法计算出一个下三角形矩阵 $\boldsymbol{G}\in\mathbb{R}^{n\times n}$ 满足 $\boldsymbol{A}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$. 对所有 $i\geqslant j, G(i,j)$ 覆盖 $A(i,j)$.

```
for j = 1 : n
    if j > 1
        A(j : n, j) = A(j : n, j) - A(j : n, 1 : j - 1)A(j, 1 : j - 1)^T
    end
```

```
        A(j:n,j) = A(j:n,j)/sqrt(A(j,j))
    end
```

本算法需 $n^2/3$ 个 flop.

### 4.2.5　基于外积的 Cholesky 分解

另一种基于外积 (秩为 1) 修正的 Cholesky 分解可通过对长方形矩阵做如下划分得到:

$$\begin{aligned} \boldsymbol{A} &= \begin{bmatrix} \alpha & \boldsymbol{v}^{\mathrm{T}} \\ \boldsymbol{v} & \boldsymbol{B} \end{bmatrix} \\ &= \begin{bmatrix} \beta & \boldsymbol{0} \\ \boldsymbol{v}/\beta & \boldsymbol{I}_{n-1} \end{bmatrix} \begin{bmatrix} 1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B} - \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}/\alpha \end{bmatrix} \begin{bmatrix} \beta & \boldsymbol{v}^{\mathrm{T}}/\beta \\ \boldsymbol{0} & \boldsymbol{I}_{n-1} \end{bmatrix}. \end{aligned} \tag{4.2.6}$$

这里 $\beta = \sqrt{\alpha}$, 且因 $\boldsymbol{A}$ 是正定的, 故 $\alpha > 0$. 而 $\boldsymbol{B} - \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}/\alpha$ 是 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 的主子矩阵, 其中

$$\boldsymbol{X} = \begin{bmatrix} 1 & -\boldsymbol{v}^{\mathrm{T}}/\alpha \\ 0 & \boldsymbol{I}_{n-1} \end{bmatrix},$$

故它也是正定的. 如果存在 Choleky 分解 $\boldsymbol{G}_1\boldsymbol{G}_1^{\mathrm{T}} = \boldsymbol{B} - \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}/\alpha$, 则由 (4.2.6) 有 $\boldsymbol{A} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$, 其中

$$\boldsymbol{G} = \begin{bmatrix} \beta & \boldsymbol{0} \\ \boldsymbol{v}/\beta & \boldsymbol{G}_1 \end{bmatrix}.$$

因此可通过反复利用 (4.2.6) 来得到最终的 Cholesky 分解, 其方式和 $kji$ 形式的高斯消去法一样.

**算法 4.2.2 (Cholesky 分解: 基于外积运算)**　给定一对称正定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 本算法计算满足 $\boldsymbol{A} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 的下三角形矩阵 $\boldsymbol{G} \in \mathbb{R}^{n\times n}$. 对所有 $i > j$, $G(i,j)$ 覆盖 $A(i,j)$.

```
for k = 1:n
    A(k,k) = sqrt(A(k,k))
    A(k+1:n,k) = A(k+1:n,k)/A(k,k)
    for j = k+1:n
        A(j:n,j) = A(j:n,j) - A(j:n,k)A(j,k)
    end
end
```

本算法需 $n^3/3$ 个 flop. 注意, 其中的 $j$ 循环计算外积的下三角部分:

$$A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k)A(k+1:n,k)^{\mathrm{T}}.$$

回想在 1.4.8 节中关于 gaxpy 运算和外积运算的比较, 容易得知算法 4.2.1 中读写向量的次数要比算法 4.2.2 少一半.

### 4.2.6 基于分块点积的 Cholesky 分解

假定 $A \in \mathbb{R}^{n\times n}$ 对称正定, 将 $A = (A_{ij})$ 和其 Cholesky 因子 $G = (G_{ij})$ 看作含有方对角块的 $N \times N$ 分块矩阵. 取出等式 $A = GG^{\mathrm{T}}$ 中关于第 $(i,j)$ 块 $(i \geqslant j)$ 的等式有

$$A_{ij} = \sum_{k=1}^{j} G_{ik} G_{jk}^{\mathrm{T}}.$$

定义

$$S = A_{ij} - \sum_{k=1}^{j-1} G_{ik} G_{jk}^{\mathrm{T}},$$

则当 $i = j$ 时, $G_{jj}G_{jj}^{\mathrm{T}} = S$; 当 $i < j$ 时, $G_{ij}G_{jj}^{\mathrm{T}} = S$. 通过合适的排序, 这些方程可用来求得所有的 $G_{ij}$.

**算法 4.2.3 (Cholesky 分解: 基于分块点积运算)** 给定一个对称正定矩阵 $A \in \mathbb{R}^{n\times n}$, 本算法可求得一个正三角矩阵 $G \in \mathbb{R}^{n\times n}$ 满足 $A = GG^{\mathrm{T}}$, $A$ 的下三角部分被 $G$ 的下三角部分覆盖, $A$ 被看做是含方对角块的 $N \times N$ 分块矩阵.

```
for j = 1 : N
    for i = j : N
        S = A_ij − Σ_{k=1}^{j−1} G_ik G_jk^T
        if i = j
            计算 Cholesky 分解 S = G_jj G_jj^T
        else
            从 G_ij G_jj^T = S 解出 G_ij
        end
        用 G_ij 覆盖 A_ij
    end
end
```

整个算法需 $n^3/3$ 个 flop, 与前述其他形式的 Cholesky 算法相同. 假定 $A$ 被适当分块, 则本算法中含有大量的矩阵乘法运算. 例如, 如果 $n = rN$, 且每个 $A_{ij}$ 都是 $r \times r$ 的, 则 3 级比例约为 $1 - (1/N^2)$.

由于没有给出积 $G_{ik}G_{jk}$ 是如何形成的以及 $r \times r$ Cholesky 分解 $S = G_{jj}G_{jj}^{\mathrm{T}}$ 是如何计算的, 因此算法 4.2.3 是不完整的. 为获得算法的高性能, 这些重要细节必须认真设计.

另一个分块算法可由基于 gaxpy 运算的 Cholesky 算法得出, 将算法 4.2.1 执行 $r$ 步后, 我们已知

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} G_{11} & 0 \\ G_{21} & I_{n-r} \end{bmatrix} \begin{bmatrix} I_r & 0 \\ 0 & \tilde{A} \end{bmatrix} \begin{bmatrix} G_{11} & 0 \\ G_{21} & I_{n-1} \end{bmatrix}^{\mathrm{T}}$$

中的矩阵 $G_{11} \in \mathbb{R}^{r\times r}$ 和 $G_{21} \in \mathbb{R}^{(n-r)\times r}$. 再接下来我们不对 $A$, 则是对已显式形成具有可利用的对称结构的约化后的矩阵 $\tilde{A} = A_{22} - G_{21}G_{21}^{\mathrm{T}}$ 进行另外的 $r$ 步基

于 gaxpy 的 Cholesky 计算. 按此方法进行下去, 我们得到基于分块的 Cholesky 分解算法, 其第 $k$ 步是对 $n-(k-1)r$ 阶的矩阵进行 $r$ 次基于 gaxpy 的 Cholesky 分解, 接着是阶为 $n-kr$ 的 3 级计算. 如果 $n \approx rN$, 则 3 级比例约为 $1-3/(2N)$.

### 4.2.7 Cholesky 分解的稳定性

在精确运算下, 我们知对称正定矩阵存在 Cholesky 分解. 反之, 如果一个 Cholesky 分解过程能够顺利进行完且得到严格大于零的平方根, 那么 $\boldsymbol{A}$ 是正定的. 因此, 判断 $\boldsymbol{A}$ 是否正定, 我们只需用上述的任一方法来试着计算其 Cholesky 分解.

有舍入误差情形是更为有意思的. Cholesky 算法的数值稳定性大致可从不等式:

$$g_{ij}^2 \leqslant \sum_{k=1}^{i} g_{ik}^2 = a_{ii}$$

导出. 该不等式说明 Cholesky 三角形矩阵因子有很好的界. 由 $\|\boldsymbol{G}\|_2^2 = \|\boldsymbol{A}\|_2$ 也可推出相同的结论.

Wilkinson (1968) 在经典论文中对 Cholesky 分解的舍入误差作了深入研究. 利用该论文之结果可证明, 如果 $\hat{x}$ 是通过上述任一 Cholesky 分解求得的 $\boldsymbol{Ax}=\boldsymbol{b}$ 的计算解, 则 $\hat{\boldsymbol{x}}$ 满足扰动方程组 $(\boldsymbol{A}+\boldsymbol{E})\hat{x}=b$, 其中 $\|\boldsymbol{E}\|_2 = c_n u\|\boldsymbol{A}\|_2, c_n$ 是由 $n$ 决定的小常数. Wilkinson 进一步指出, 如果 $q_n u\kappa_2(\boldsymbol{A}) \leqslant 1$, 其中 $q_n$ 是另一个小常数, 则 Cholesky 分解能够执行到底, 而不会出现对负数开平方根.

**例 4.2.2** 如果用算法 4.2.2 来处理下述正定矩阵:

$$\boldsymbol{A} = \begin{bmatrix} 100 & 15 & 0.01 \\ 15 & 2.3 & 0.01 \\ 0.01 & 0.01 & 1.00 \end{bmatrix},$$

且用 $\beta=10, t=2$ 的有限位计算, 则 $\hat{g}_{11} =_{10}, \hat{g}_{21}=1.5, \hat{g}_{31}=0.001, \hat{g}_{22}=0.00$. 本算法在求解 $g_{32}$ 时失败.

### 4.2.8 半定矩阵

如果对所有的向量 $\boldsymbol{x}$ 都有 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Ax} \geqslant 0$, 则称矩阵 $\boldsymbol{A}$ 是半正定的. 对称半正定 (sps) 矩阵也是一类非常重要的矩阵, 我们仅简单地讨论一些可用来求解 sps 问题的 Cholesky 型方法. 首先需要关于 sps 矩阵对角元的一些结论.

**定理 4.2.6** 如果 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是对称半正定的, 则

$$|a_{ij}| \leqslant (a_{ii}+a_{jj})/2, \tag{4.2.7}$$

$$|a_{ij}| \leqslant \sqrt{a_{ii}a_{jj}} \quad (i \neq j), \tag{4.2.8}$$

$$\max_{i,j}|a_{ij}| = \max_{i} a_{ii}, \tag{4.2.9}$$

$$a_{ii}=0 \Rightarrow A(i,:)=0, \quad A(:,i)=0. \tag{4.2.10}$$

**证明** 如果 $\boldsymbol{x}=\boldsymbol{e}_i+\boldsymbol{e}_j$, 则 $0 \leqslant \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}=a_{ii}+a_{jj}+2a_{ij}$, 而如果 $\boldsymbol{x}=\boldsymbol{e}_i-\boldsymbol{e}_j$, 则 $0 \leqslant \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}=a_{ii}+a_{jj}-2a_{ij}$. 不等式 (4.2.7) 可由此二式推出. 等式 (4.2.9) 是 (4.2.7) 的直接推论.

为证 (4.2.8), 不失一般性, 令 $i=1, j=2$, 考察不等式

$$0 \leqslant \begin{bmatrix} x \\ 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = a_{11}x^2+2a_{12}x+a_{22}.$$

由于 $A(1:2,1:2)$ 也是半正定的, 因此上式成立. 这是一个关于 $x$ 的二次方程, 为使不等号成立, 需判别式 $4a_{12}^2-4a_{11}a_{22}$ 小于零. 由 (4.2.8) 可推出 (4.2.10). □

考察基于外积的 Cholesky 分解应用于 sps 矩阵时将会出现什么结果. 如果元素 $A(k,k)$ 为零, 则由 (4.2.10) 知 $A(k:n,k)$ 为零, 故不进行任何操作, 这样有

```
for k = 1 : n
    if A(k,k) > 0
        A(k,k) = sqrt(A(k,k))
        A(k+1:n,k) = A(k+1:n,k)/A(k,k)
        for j = k+1 : n
            A(j:n,j) = A(j:n,j) - A(j:n,k)A(j,k)        (4.2.11)
        end
    end
end
```

于是, 对算法 4.2.2 作简单修改就可以适用于半正定矩阵. 然而, 实际计算中由于舍入而产生误差, 可能导致结果不会恰好为零, 所以最好是进行选主元.

### 4.2.9 对称的选主元

为保持对称矩阵 $\boldsymbol{A}$ 的对称性, 我们仅考虑形如 $\boldsymbol{PAP}^{\mathrm{T}}$ 的数据重排列, 其中 $\boldsymbol{P}$ 是置换矩阵. 仅做行置换 $(\boldsymbol{A} \leftarrow \boldsymbol{PA})$ 或列置换 $(\boldsymbol{A} \leftarrow \boldsymbol{AP})$ 都会破坏对称性. 形如

$$\boldsymbol{A} \leftarrow \boldsymbol{PAP}^{\mathrm{T}}$$

的修正称为 $\boldsymbol{A}$ 的*对称置换*. 这种操作不会将非对角元移到对角位置上. $\boldsymbol{PAP}^{\mathrm{T}}$ 的对角元素是 $\boldsymbol{A}$ 的对角元的一个重新排序.

假定在 (4.2.11) 的第 $k$ 步的开始我们将 $A(k:n,k:n)$ 中最大的对角元对称置换到主位置上. 如果此最大对角元为零, 则由 4.2.10 知 $A(k:n,k:n)=0$. 这样我们可以计算分解 $\boldsymbol{PAP}^{\mathrm{T}}=\boldsymbol{GG}^{\mathrm{T}}$, 其中 $\boldsymbol{G} \in \mathbb{R}^{n \times (k-1)}$ 是下三角形矩阵.

**算法 4.2.4** 假定 $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ 是对称半正定矩阵, 且秩为 $r$, 本算法求得满足 $\boldsymbol{PAP}^{\mathrm{T}}=\boldsymbol{GG}^{\mathrm{T}}$ 的 $n \times r$ 下三角形矩阵 $\boldsymbol{G}$, 其中 $\boldsymbol{P}$ 是置换矩阵, $r$ 是下标. $A(:,1:r)$

的下三角部分被 $G$ 的下三角部分所覆盖, $P = P_r \cdots P_1$, 其中 $P_k$ 是将单位矩阵的第 $k$ 行与第 piv($k$) 行互换.

$r = 0$
**for** $k = 1:n$
  找 $q(k \leqslant q \leqslant n)$ 使得 $A(q,q) = \max\{A(k,k),\cdots,A(n,n)\}$
  **if** $A(q,q) > 0$
    $r = r + 1$
    $\text{piv}(k) = q$
    $A(k,:) \leftrightarrow A(q,:)$
    $A(:,k) = A(:,q)$
    $A(k,k) = \sqrt{A(k,k)}$
    $A(k+1:n,k) = A(k+1:n,k)/A(k,k)$
    **for** $j = k+1:n$
      $A(j:n,j) = A(j:n,j) - A(j:n,k)A(j,k)$
    **end**
  **end**
**end**

在实际计算中, 使用了容差来检测小的 $A(k,k)$ 值. 但这种处理需要很高的技巧性, 读者可参考 Higham (1989). 此外, 5.5 节中对秩检测问题的容差作了讨论. 最后我们指出算法 4.2.4 的真正高效实现将仅需要 $A$ 的下三角部分.

### 4.2.10 极分解与平方根

设 $A = U_1\Sigma_1V^{\mathrm{T}}$ 是 $A \in \mathbb{R}^{m\times n}$ 的窄 SVD 分解, 其中 $m \geqslant n$. 注意到

$$A = (U_1V^{\mathrm{T}})(V\Sigma_1V^{\mathrm{T}}) \equiv ZP, \tag{4.2.12}$$

其中 $Z = U_1V^{\mathrm{T}}, P = V\Sigma_1V^{\mathrm{T}}$. 由

$$x^{\mathrm{T}}Px = (V^{\mathrm{T}}x)^{\mathrm{T}}\Sigma_1(V^{\mathrm{T}}x) = \sum_{k=1}^{n}\sigma_k y_k^2 \geqslant 0,$$

其中 $y = V^{\mathrm{T}}x$, 知 $Z$ 的列正交, $P$ 是对称半正定的. (4.2.12) 称为**极分解**, 因为它与复数分解 $z = \mathrm{e}^{\mathrm{i}\cdot\arg(z)}|z|$ 类似. 更详细的内容参见 12.4.1 节.

另一类重要的分解是**矩阵平方根**. 假定 $A \in \mathbb{R}^{n\times n}$ 是对称半正定的且 $A = GG^{\mathrm{T}}$ 是其 Cholesky 分解. 如果 $G = U\Sigma V^{\mathrm{T}}$ 是 $G$ 的 SVD 分解, $X = U\Sigma U^{\mathrm{T}}$, 则 $X$ 是对称半正定的, 且有

$$A = GG^{\mathrm{T}} = (U\Sigma V^{\mathrm{T}})(U\Sigma V^{\mathrm{T}})^{\mathrm{T}} = U\Sigma^2U^{\mathrm{T}} = (U\Sigma U^{\mathrm{T}})(U\Sigma U^{\mathrm{T}}) = X^2.$$

因此 $X$ 是 $A$ 的**平方根**. 可以证明 (用特征值理论很容易), 对称半正定矩阵有一个唯一的对称半正定的平方根.

## 习 题

**4.2.1** 假定 $\boldsymbol{H}=\boldsymbol{A}+\mathrm{i}\boldsymbol{B}$ 是 Hermit 矩阵且正定，其中 $\boldsymbol{A},\boldsymbol{B}\in\mathbb{R}^{n\times n}$，这意味着 $\boldsymbol{x}\neq\boldsymbol{0}$ 时，$\boldsymbol{x}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}>0$

(a) 证明：$\boldsymbol{C}=\begin{bmatrix}\boldsymbol{A} & -\boldsymbol{B}\\ \boldsymbol{B} & \boldsymbol{A}\end{bmatrix}$ 是对称正定的.

(b) 设计一算法求解 $(\boldsymbol{A}+\mathrm{i}\boldsymbol{B})(\boldsymbol{x}+\mathrm{i}\boldsymbol{y})=(\boldsymbol{b}+\mathrm{i}\boldsymbol{c})$，其中 $\boldsymbol{b},\boldsymbol{c},\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是 $\mathbb{R}^n$ 上的，该算法应在 $8n^3/3$ 个 flop 内完成，并指出所需的内存空间.

**4.2.2** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称正定矩阵，给出一个算法来求解满足 $\boldsymbol{A}=\boldsymbol{R}\boldsymbol{R}^{\mathrm{T}}$ 的上三角形矩阵 $\boldsymbol{R}\in\mathbb{R}^{n\times n}$.

**4.2.3** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称正定矩阵，令 $\boldsymbol{T}=(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})/2,\boldsymbol{S}=(\boldsymbol{A}-\boldsymbol{A}^{\mathrm{T}})/2$.

(a) 证明 $\|\boldsymbol{A}^{-1}\|_2\leqslant\|\boldsymbol{T}^{-1}\|_2$ 和对于所有 $\boldsymbol{x}\in\mathbb{R}^n$ 有 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{x}\leqslant\boldsymbol{x}^{\mathrm{T}}\boldsymbol{T}^{-1}\boldsymbol{x}$ 成立.

(b) 证明如果 $\boldsymbol{A}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{M}^{\mathrm{T}}$，则对 $k=1:n$ 有 $d_k\geqslant 1/\|\boldsymbol{T}^{-1}\|_2$.

**4.2.4** 寻找一个 $2\times 2$ 的实矩阵 $A$，使其所有的实非零 2 维向量都具有性质 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}>0$，但在 $\mathbb{C}^{2\times 2}$ 范围内它却不是正定的.

**4.2.5** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 有正对角元，证明如果 $\boldsymbol{A}$ 和 $\boldsymbol{A}^{\mathrm{T}}$ 都是严格对角占优的，则 $\boldsymbol{A}$ 是正定的.

**4.2.6** 证明函数 $f(\boldsymbol{x})=(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})/2$ 是 $\mathbb{R}^n$ 上的向量范数，当且仅当 $\boldsymbol{A}$ 是正定的.

**4.2.7** 修改算法 4.2.1 使得如果出现负数的平方根的情形，算法找到一个单位向量 $\boldsymbol{x}$，使满足 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}<0$ 后算法终止.

**4.2.8** 复矩阵 $\boldsymbol{A}$ 的数值域 $W(\boldsymbol{A})$ 定义为：$W(\boldsymbol{A})=\{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{x}:\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}=1\}$，证明如果 $0\notin W(\boldsymbol{A})$，则 $\boldsymbol{A}$ 有 LU 分解.

**4.2.9** 给出 $\boldsymbol{A}\in\mathbb{R}^{m\times n}(m<n)$ 的一个极分解.

**4.2.10** 假定 $\boldsymbol{A}=\boldsymbol{I}+\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}$，其中 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 且 $\|\boldsymbol{u}\|_2=1$，给出 $\boldsymbol{A}$ 的 Cholesky 因子的对角元和次对角元的显式表达式.

**4.2.11** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称正定矩阵且其 Cholesky 因子存在，令 $e_k=I_n(:,k)$，对于 $1\leqslant i<j\leqslant n$，令 $\alpha_{ij}$ 是使得 $\boldsymbol{A}+\alpha(\boldsymbol{e}_i\boldsymbol{e}_j^{\mathrm{T}}+\boldsymbol{e}_j\boldsymbol{e}_i^{\mathrm{T}})$ 奇异的最小实数. 同样，令 $a_{ij}$ 是使得 $(\boldsymbol{A}+\alpha\boldsymbol{e}_i\boldsymbol{e}_i^{\mathrm{T}})$ 奇异的最小实数. 给出应用 Sherman-Morrison-Woodbury 公式求解这些量的方法，并指出求解全部 $\alpha_{ij}$ 的 flop 数.

## 本节注释与参考文献

下述问题的数学运算经常能给出二次型 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$ 的正定形式. 例如，一些偏微分算子的离散化所产生的矩阵可被证明是正定的. 下面的文献讨论了非对称的正定问题：

A. Buckley(1974). "A Note on Matrices $A=I+H,H$ Skew-Symmetric," *Z. Angew. Math. Mech.* 54, 125–126.

A. Buckley (1977). "On the Solution of Certain Shew-Symmetric Linear Systems," *SIAM J. Num. Anal.* 14, 566–570.

G. H. Golub and C. Van Loan (1979). "Unsymmetric Positive Definite Linear Systems," *Lin. Alg. and Its Applic.* 28, 85–98.

R. Mathias (1992). "Matrices with Positive Definite Hermitian Part: Inequalities and Linear Systems," *SIAM J. Matrix Anal. Appl.* 13, 640–654.

对称正定方程组是特殊 $Ax = b$ 问题的最重要的一类, 下述文献给出了解决这些问题的算法:

R. S. Martin, G. Peters, and J. H. Wilkinson (1965). "Symmetric Decomposition of a Positive Definite Matrix," *Numer. Math.* 7, 362–383.

R. S. Martin, G. Peters, and J. H. Wilkinson (1966). "Iterative Refinement of the Solution of a Positive Definite System of Equations," *Numer. Math.* 8, 203–216.

F. L. Bauer and C. Reinsch (1971). "Inversion of Positive Definite Matrices by the Gauss-Jordan Method," in *Handbook for Automatic Computation Vol. 2, Linear Algebra*, J. H. Wilkinson and C. Reinsch, eds. Springer-Verlag, New York, 45–49.

下述文献分析了此方法的舍入误差:

J. H. Wilkinson (1968). "À Priori Error Analysis of Algebraic Processes," *Proc. International Congress Math.* (Moscow: Izdat. Mir, 1968), pp. 629–639.

J. Meinguet (1983). "Refined Error Analyses of Cholesky Factorization," *SIAM J. Numer. Anal.* 20, 1243–1250.

A. Kielbasinski (1987). "A Note on Rounding Error Analysis of Cholesky Factorization," *Lin. Alg. and Its Applic.* 88/89, 487–494.

N. J. Higham (1990). "Analysis of the Cholesky Decomposition of a Semidefinite Matrix," in *Reliable Numerical Computation,* M. G. Cox and S. J. Hammarling (eds), Oxford University Press, Oxford, UK, 161–185.

R. Carter (1991), "Y-MP Floating Point and Cholesky Factorization," *Int'l J. High Speed Computing* 3, 215–222.

J-Guang Sun (1992). "Rounding Error and Perturbation Bounds for the Cholesky and $LDL^T$ Factorizations," *Lin. Alg. and Its Applic.* 173, 77–97.

当 $A = GG^T$ 受到扰动时 Cholesky 三角形矩阵 $G$ 如何变化, 下述文献给出这个问题的分析:

G. W. Stewart (1977b). "Perturbation Bounds for the $QR$ Factorization of a Matrix," *SIAM J. Num. Anal.* 14, 509–518.

Z. Dramăc, M. Omladič, and K. Veselič (1994). "On the Perturbation of the Cholesky Factorization," *SIAM J. Matrix Anal. Appl.* 15, 1319–1332.

正半定矩阵的敏感性问题和极分解的敏感性问题在下述文献中给予了讨论.

N. J. Higham(1988). "Computing a Nearest Symmetric Positive Semidefinite Matrix," *Lin. Alg. and Its Applic.* 103, 103–118.

R. Mathias (1993). "Perturbation Bounds for the Polar Decomposition," *SIAM J. Matrix Anal. Appl.* 14, 588–597.

R-C. Li (1995). "New Perturbation Bounds for the Unitary Polar Factor," *SIAM J. Matrix Anal. Appl.* 16, 327–332.

关于极分解和平方根的计算方面的参考文献分别在 8.6 节和 11.2 节中给出.

## 4.3 带状方程组

在涉及线性方程组的许多应用中, 系数矩阵是带状的, 只要方程排序能使得每个未知数 $x_i$ 只出现在与第 $i$ 个方程相邻的几个方程中就会出现这种情况. 严格地说, 如果对 $j > i+q$ 有 $a_{ij}=0$, 则称 $\boldsymbol{A}=(a_{ij})$ 具有**上带宽** $q$; 如果对 $i>j+p$ 有 $a_{ij}=0$, 则称 $\boldsymbol{A}=(a_{ij})$ 具有**下带宽** $p$. 当求解带状方程组时, 由于 LU, $\mathrm{GG^T}$, $\mathrm{LDM^T}$ 等中的三角因子也是带状的, 因此可节省很多.

开始本节前, 建议读者复习一下 1.2 节中关于带状矩阵乘法的内容.

### 4.3.1 带状矩阵的 LU 分解

首先要给出的结论是, 如 $\boldsymbol{A}$ 是带状矩阵且有 LU 分解 $\boldsymbol{A}=\boldsymbol{LU}$, 则 $\boldsymbol{L}(\boldsymbol{U})$ 具有与 $\boldsymbol{A}$ 相同的下 (上) 带宽.

**定理 4.3.1** *假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 有 LU 分解 $\boldsymbol{A}=\boldsymbol{LU}$, 如果 $\boldsymbol{A}$ 的上带宽为 $q$, 下带宽为 $p$, 则 $\boldsymbol{U}$ 的上带宽为 $q, \boldsymbol{L}$ 的下带宽为 $p$.*

**证明** 证明过程是对 $n$ 做归纳法. 由 (3.2.6) 有如下分解:

$$\boldsymbol{A}=\begin{bmatrix}\alpha & \boldsymbol{\omega}^{\mathrm T}\\ \boldsymbol{v} & \boldsymbol{B}\end{bmatrix}=\begin{bmatrix}1 & \boldsymbol{0}\\ \boldsymbol{v}/\alpha & \boldsymbol{I}_{n-1}\end{bmatrix}\begin{bmatrix}1 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{B}-\boldsymbol{v}\boldsymbol{\omega}^{\mathrm T}/\alpha\end{bmatrix}\begin{bmatrix}\alpha & \boldsymbol{\omega}^{\mathrm T}\\ \boldsymbol{0} & \boldsymbol{I}_{n-1}\end{bmatrix}.$$

很显然只有 $\boldsymbol{\omega}$ 的前 $q$ 个分量和 $\boldsymbol{v}$ 的前 $p$ 个分量是非零的, 故 $\boldsymbol{B}-\boldsymbol{v}\boldsymbol{\omega}^{\mathrm T}/\alpha$ 的上带宽为 $q$, 下带宽为 $q$. 设 $\boldsymbol{L}_1\boldsymbol{U}_1$ 是此矩阵的 LU 分解, 应用归纳假设和 $\boldsymbol{\omega},\boldsymbol{v}$ 的稀疏性得

$$\boldsymbol{L}=\begin{bmatrix}1 & \boldsymbol{0}\\ \boldsymbol{v}/\alpha & \boldsymbol{L}_1\end{bmatrix},\quad \boldsymbol{U}=\begin{bmatrix}\alpha & \boldsymbol{\omega}^{\mathrm T}\\ \boldsymbol{0} & \boldsymbol{U}_1\end{bmatrix}$$

均具有欲证的带宽且满足 $\boldsymbol{A}=\boldsymbol{LU}$. □

针对带状矩阵的专用高斯消去法存在 LU 分解也是一目了然的.

**算法 4.3.1 (带状高斯消去法: 基于外积的形式)** *给定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 的上带宽为 $q$, 下带宽为 $p$, 当其有 LU 分解时, 本算法计算此分解. 当 $i>j$ 时, $A(i,j)$ 被 $L(i,j)$ 覆盖, 否则被 $U(i,j)$ 覆盖.*

```
for k = 1 : n − 1
    for i = k + 1 : min(k + p, n)
        A(i, k) = A(i, k)/A(k, k)
    end
    for j = k + 1 : min(k + q, n)
        for i = k + 1 : min(k + p, n)
            A(i, j) = A(i, j) − A(i, k)A(k, j)
        end
    end
```

```
    end
```

如果 $n \gg p$ 且 $n \gg q$, 则本算法需要 $2npq$ 个 flop. 算法 4.1.1 ($\mathrm{LDM^T}$) 及所有的 Cholesky 分解算法都同样有带状的形式, 留给读者练习.

### 4.3.2 带状三角方程组的求解

求解带状三角方程组时也可类似地节约很多工作量.

**算法 4.3.2 (带状向前消去法: 列形式)** 令 $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ 是下带宽为 $p$ 的单位下三角形矩阵, 给定 $\boldsymbol{b} \in \mathbb{R}^n$, 本算法以 $\boldsymbol{Lx} = \boldsymbol{b}$ 的解覆盖 $\boldsymbol{b}$.

```
for j = 1 : n
    for i = j + 1 : min(j + p, n)
        b(i) = b(i) − L(i, j)b(j)
    end
end
```

如果 $n \geqslant p$, 该算法需要 $2np$ 个 flop.

**算法 4.3.3 (带状向后消去法: 行形式)** 令 $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ 是上带宽为 $q$ 的非奇异上三角形矩阵. 给定 $\boldsymbol{b} \in \mathbb{R}^n$, 本算法以 $\boldsymbol{Ux} = \boldsymbol{b}$ 之解覆盖 $\boldsymbol{b}$.

```
for j = n : −1 : 1
    b(j) = b(j)/U(j, j)
    for i = max(1, j − q) : j − 1
        b(i) = b(i) − U(i, j)b(j)
    end
end
```

如果 $n \gg q$, 该算法需要 $2nq$ 个 flop.

### 4.3.3 选主元的带状高斯消去法

列选主元的高斯消去法也可利用 $\boldsymbol{A}$ 的带状结构进行特殊修改. 然而, 如果 $\boldsymbol{PA} = \boldsymbol{LU}$, 那么 $\boldsymbol{L}$ 和 $\boldsymbol{U}$ 的带状性质却不那么简单. 例如, 如果 $\boldsymbol{A}$ 是三对角矩阵, 在算法执行的第一步将前两行互换后, $u_{13}$ 就非零了. 结果, 行的互换将带宽扩大了. 精确地知道带宽如何扩大是下面定理的主题.

**定理 4.3.2** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 非奇异, 其上、下带宽分别为 $q$ 和 $p$. 如果用列选主元的高斯消去法来计算高斯变换

$$\boldsymbol{M}_j = \boldsymbol{I} - \alpha^{(j)}\boldsymbol{e}_j^{\mathrm{T}}, \quad j = 1 : n-1$$

和置换矩阵 $\boldsymbol{P}_1, \cdots, \boldsymbol{P}_{n-1}$ 使得 $\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A} = \boldsymbol{U}$ 是上三角形矩阵, 则 $\boldsymbol{U}$ 的上带宽为 $p+q$, 而且当 $i \leqslant j$ 或 $i > j+p$ 时, $\alpha^{(j)} = 0$.

**证明** 设 $\boldsymbol{PA} = \boldsymbol{LU}$ 是列选主元高斯消去法所计算的分解, 由前知 $\boldsymbol{P} = \boldsymbol{P}_{n-1}\cdots\boldsymbol{P}_1$. 将 $\boldsymbol{P}^{\mathrm{T}}$ 写成 $\boldsymbol{P}^{\mathrm{T}} = [\boldsymbol{e}_{s_1}, \cdots, \boldsymbol{e}_{s_n}]$, 其中 $\{s_1, \cdots, s_n\}$ 是 $\{1, 2, \cdots, n\}$ 的

一个置换. 如果 $s_i > i+p$, 则 $\boldsymbol{PA}$ 的 $i$ 阶顺序主子矩阵是奇异的, 这是因为对于 $j=1:s_i-p-1$ 有 $(\boldsymbol{PA})_{ij}=a_{s_i,j}=0$ 而且 $s_i-p-1 \geqslant i$. 于是可推出 $\boldsymbol{U}$ 和 $\boldsymbol{A}$ 是奇异的, 这是一个矛盾. 因此对于 $i=1:n$, 有 $s_i \leqslant i+p$, 因此 $\boldsymbol{PA}$ 的上带宽为 $p+q$. 由定理 4.3.1 知 $\boldsymbol{U}$ 的上带宽为 $p+q$.

可通过观察 $M_j$ 来证实 $\alpha^{(j)}$ 的性质, 因 $M_j$ 的形成只需要将已部分约化的矩阵 $\boldsymbol{P}_j\boldsymbol{M}_{j-1}\boldsymbol{P}_{j-1}\cdots\boldsymbol{P}_1\boldsymbol{A}$ 中的 $(j+1,j),\cdots,(j+p,j)$ 元素化为零. □

因此就 $\boldsymbol{U}$ 变得比 $\boldsymbol{A}$ 的上三角形矩阵 "更宽" 这一点来说, 选主元会破坏带状结构, 然而关于 $\boldsymbol{L}$ 的带宽没有任何结果. 但由于 $\boldsymbol{L}$ 的第 $j$ 列是高斯向量 $\alpha_j$ 的一个置换, 因此 $\boldsymbol{L}$ 的每一列至多只有 $p+1$ 个非零元.

### 4.3.4 Hessenberg LU 分解

做为非对称带状矩阵计算的一个例子, 我们看看列选主元的高斯消去法是如何分解一个上 Hessenberg 矩阵 $\boldsymbol{H}$ 的 (回忆一下, 如果 $\boldsymbol{H}$ 是上 Hessenberg 矩阵, 则当 $i>j+1$ 时 $h_{ij}=0$). 经过 $k-1$ 步的列选主元的高斯消去法, 得到如下形式的一个上 Hessenberg 矩阵:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix} \quad k=3, \quad n=5.$$

由这个矩阵的特殊结构可知下一个置换矩阵 $\boldsymbol{P}_3$ 是单位矩阵或者是交换第 3 行和第 4 行的单位矩阵. 而且下一步的高斯变换 $\boldsymbol{M}_k$ 在位置 $(k+1,k)$ 有一个非零乘子. 这刻画了下面算法的第 $k$ 步.

**算法 4.3.4 (Hessenberg LU)** 给出一个 Hessenberg 矩阵 $\boldsymbol{H}\in\mathbb{R}^{n\times n}$, 本算法计算出上三角形矩阵 $\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{H}=\boldsymbol{U}$, 其中每个 $\boldsymbol{P}_k$ 是置换矩阵, 每个 $\boldsymbol{M}_k$ 是元素以 1 为界的高斯变换. $H(i,k)$ 在 $i\leqslant k$ 时由 $U(i,k)$ 覆盖, 在 $i=k+1$ 时由 $(\boldsymbol{M}_k)_{k+1,k}$ 覆盖. 用整数向量 $\mathrm{piv}(1:n-1)$ 来记录置换. 如果 $\boldsymbol{P}_k=\boldsymbol{I}$, 则 $\mathrm{piv}(k)=0$. 如果 $\boldsymbol{P}_k$ 是将第 $k$ 行和第 $k+1$ 行互换, 则 $\mathrm{piv}(k)=1$.

```
for k = 1 : n − 1
    if |H(k, k)| < |H(k + 1, k)|
        piv(k) = 1; H(k, k : n) ↔ H(k + 1, k : n)
    else
        piv(k) = 0
    end
    if H(k, k) ≠ 0
        t = −H(k + 1, k)/H(k, k)
```

```
            for j = k+1 : n
                H(k+1, j) = H(k+1, j) + tH(k, j)
            end
            H(k+1, k) = t
        end
    end
```

该算法需要 $n^2$ 个 flop.

### 4.3.5　带状 Cholesky 分解

本节的剩下部分考虑当 $\boldsymbol{A}$ 是对称正定的带状矩阵时, 如何求解 $\boldsymbol{A}x = b$. 由于这种情形不需要进行选主元, 因此可写出许多紧凑优美的算法. 特别是, 由定理 4.3.1, 如果 $\boldsymbol{A} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 是 $\boldsymbol{A}$ 的 Cholesky 分解, 则 $\boldsymbol{G}$ 的下宽带与 $\boldsymbol{A}$ 相同. 这导致如下算法, 它是算法 4.2.1 (基于 gaxpy 的 Cholesky 分解) 的带状形式.

**算法 4.3.5 (带状 Cholesky 分解: gaxpy 形式)**　给定一个带宽为 $p$ 的对称正定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 本算法计算一个下带宽为 $p$ 的下三角形矩阵 $\boldsymbol{G}$, 使得 $\boldsymbol{A} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$. 对所有的 $i \geqslant j, A(i,j)$ 被 $G(i,j)$ 覆盖.

```
for j = 1 : n
    for k = max(1, j-p) : j-1
        λ = min(k+p, n)
        A(j:λ, j) = A(j:λ, j) - A(j,k)A(j:λ, k)
    end
    λ = min(j+p, n)
    A(j:λ, j) = A(j:λ, j)/sqrt(A(j,j))
end
```

如果 $n \gg p$, 则此算法需 $n(p^2+3p)$ 个 flop 和 $n$ 次平方根运算. 当然, 在真正实现时应为 $\boldsymbol{A}$ 设计恰当的数据结构. 例如, 如果仅存储 $\boldsymbol{A}$ 的非零下三角部分, 则一个 $(p+1)\times n$ 的数组就足够了 (见 1.2.6 节).

如果将带状 Cholesky 分解与适当的带状三角方程组求解算法结合, 则解整个 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 问题大约需 $np^2+7np+2n$ 个 flop 和 $n$ 次平方根运算. 当 $p$ 值不大时, 求根运算在整个计算中占很大比例. 故最好是用 $\mathrm{LDL}^{\mathrm{T}}$ 分解. 实际上, 仔细的计数可发现, 利用 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}, \boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}, \boldsymbol{D}\boldsymbol{z} = \boldsymbol{y}$ 和 $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{z}$ 来解决问题只需要 $np^2+8np+n$ 个 flop 而且不必进行平方根运算.

### 4.3.6　三对角方程组的求解

作为求解窄的带状矩阵 $\mathrm{LDL}^{\mathrm{T}}$ 问题的例子, 我们来考察对称正定三对角方程组. 设

$$L = \begin{bmatrix} 1 & & & \cdots & 0 \\ e_1 & 1 & & & \vdots \\ & \ddots & \ddots & & \\ \vdots & & \ddots & & \\ 0 & \cdots & & e_{n-1} & 1 \end{bmatrix}$$

和 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n)$, 则由等式 $\boldsymbol{A} = \boldsymbol{LDL}^{\mathrm{T}}$ 得出

$$\begin{aligned} a_{11} &= d_1 & & \\ a_{k,k-1} &= e_{k-1}d_{k-1} & & k = 2:n \\ a_{kk} &= d_k + e_{k-1}^2 d_{k-1} = d_k + e_{k-1}a_{k,k-1} & & k = 2:n \end{aligned}$$

于是 $d_i$ 和 $e_i$ 可这样求出:

$d_1 = a_{11}$
**for** $k = 2:n$
  $e_{k-1} = a_{k,k-1}/d_{k-1}; d_k = a_{kk} - e_{k-1}a_{k,k-1}$
**end**

可通过解 $\boldsymbol{Ly} = \boldsymbol{b}, \boldsymbol{Dz} = \boldsymbol{y}$ 和 $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{z}$ 来得到 $\boldsymbol{Ax} = \boldsymbol{b}$ 的解. 利用覆盖, 我们可得到下列算法.

**算法 4.3.6 (对称三对角正定方程组的解法)** 给定一个 $n \times n$ 三对角对称正定矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{b} \in \mathbb{R}^n$, 本算法以 $\boldsymbol{Ax} = \boldsymbol{b}$ 的解来覆盖 $b$. 我们假定 $\boldsymbol{A}$ 的对角线元素存储在 $d(1:n)$ 中, 次对角线元素存储在 $e(1:n-1)$ 中.

**for** $k = 2:n$
  $t = e(k-1); e(k-1) = t/d(k-1); d(k) = d(k) - te(k-1)$
**end**
**for** $k = 2:n$
  $b(k) = b(k) - e(k-1)b(k-1)$
**end**
$b(n) = b(n)/d(n)$
**for** $k = n-1:-1:1$
  $b(k) = b(k)/d(k) - e(k)b(k+1)$
**end**

该算法需要 $8n$ 个 flop.

### 4.3.7 向量化问题

三对角的例子引发出一个难题: 即求解窄的带状问题与处理机的向量/流水线结构不相适. 窄的带状意味着短向量. 然而有时候却需要同时解决大量的且相互独立的这类问题. 让我们以 1.4 节中提出的观点来看应如何安排计算.

为清晰起见, 假设我们求解 $n \times n$ 单位下双对角方程组

$$A^{(k)}x^{(k)} = b^{(k)}, \quad k = 1:m$$

且 $m \gg n$. 假定有数组 $E(1:n-1, 1:m)$ 和 $B(1:n, 1:m)$, 用 $E(1:n-1, k)$ 来存储 $\boldsymbol{A}^{(k)}$ 的次对角线, $B(1:n, k)$ 来存储右端向量 $b^{(k)}$ 的第 $k$ 个分量. 我们按下述方式来用解 $x^{(k)}$ 覆盖 $b^{(k)}$

```
for k = 1 : m
    for i = 2 : n
        B(i,k) = B(i,k) - E(i-1,k)B(i-1,k)
    end
end
```

本算法是按顺序求解每个双对角方程组的, 引起的问题是内层循环没有向量化. 这是由于 $B(i,k)$ 对 $B(i-1,k)$ 的依赖造成的. 将关于 $k$ 和 $i$ 的循环互换就得到:

```
for i = 2 : n
    for k = 1 : m
        B(i,k) = B(i,k) - E(i-1,k)B(i-1,k)        (4.3.1)
    end
end
```

这样内层循环已经很好地向量化, 因为它由一次向量乘法和一次向量加法组成. 不幸的是, (4.3.1) 不是整体流运算. 但是, 这个问题可通过按行存储次对角元和右端向量来解决. 即采用数组 $E(1:m, 1:n-1)$ 和 $B(1:m, 1:n-1)$, 用 $E(k, 1:n-1)$ 来存储 $\boldsymbol{A}$ 的次对角元, 用 $B(k, 1:n)$ 来存储 $\boldsymbol{b}^{(k)\mathrm{T}}$. (4.3.1) 的计算过程则变为

```
for i = 2 : n
    for k = 1 : m
        B(k,i) = B(k,i) - E(k,i-1)B(k,i-1)
    end
end
```

这再次说明数据结构对计算的影响.

### 4.3.8 带状矩阵的数据结构

上面算法的写法基于 $\boldsymbol{A}$ 按常规方式储存于 $n \times n$ 数组中. 在实际应用中, 可针对带状线性方程组的求解有效地设计特殊的数据结构, 以充分利用矩阵中许多元素为零的特点. 回想 1.2.6 节的内容即知, 如果 $\boldsymbol{A}$ 的下带宽为 $p$, 上带宽为 $q$, 则矩阵可以用一个 $(p+q+1) \times n$ 数组 A.band 来存储, 其中位于带上的元素 $a_{ij}$ 存于 A.band$(i-j+q+1, j)$ 中. 按这种方式, A.band 的第 $j$ 列存储 $\boldsymbol{A}$ 的第 $j$ 列的非零部分. 1.2.8 节中讨论的另一种矩阵数据结构是用一个一维数组 A.diag 将 $\boldsymbol{A}$ 按对

角线存储. 不管采用哪种数据结构, 涉及带状阵存储的矩阵运算都要尽量减少下标换算的开销.

## 习 题

**4.3.1** 导出一个类似于算法 4.3.1 的带状 $\mathrm{LDM^T}$ 分解算法.

**4.3.2** 说明如何将算法 4.3.4 产生的结果应用于解上 Hessenberg 方程组 $\boldsymbol{Hx}=\boldsymbol{b}$.

**4.3.3** 给出一个用列选主元的高斯消去法来求解非对称三对角方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的算法, 要求只使用 4 个 $n$ 阶浮点数向量的存储空间.

**4.3.4** 对于 $\boldsymbol{C}\in\mathbb{R}^{n\times n}$ 定义形状指标 $m(\boldsymbol{C},i)=\min\{j:c_{ij}\neq 0\}$, 其中 $i=1:n$, 证明如果 $\boldsymbol{A}=\boldsymbol{GG}^{\mathrm T}$ 是 $\boldsymbol{A}$ 的 Cholesky 分解, 则对于 $i=1:n$ 有 $m(A,i)=m(\boldsymbol{G},i)$. (我们称 $\boldsymbol{G}$ 和 $\boldsymbol{A}$ 有相同的形状.)

**4.3.5** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称正定矩阵且有 $m_i=m(\boldsymbol{A},i)$, 其中 $i=1:n$. 假定 $\boldsymbol{A}$ 存储于一维数组 $v$ 中: $v=(a_{11},a_{2,m_2},\cdots,a_{22},a_{3,m_3},\cdots,a_{n,m_n},\cdots,a_{nn})$. 编写一个算法, 用 Cholesky 因子 $\boldsymbol{G}$ 中对应的元素去覆盖 $v$, 用这个分解结果解 $\boldsymbol{Ax}=\boldsymbol{b}$, 问需多少个 flop?

**4.3.6** 对于 $\boldsymbol{C}\in\mathbb{R}^{n\times n}$ 定义 $p(\boldsymbol{C},i)=\max\{j:c_{ij}\neq 0\}$, 假设 $\boldsymbol{A}$ 有 LU 分解 $\boldsymbol{A}=\boldsymbol{LU}$ 且有

$$m(\boldsymbol{A},1)\leqslant m(\boldsymbol{A},2)\leqslant\cdots\leqslant m(\boldsymbol{A},n),$$
$$p(\boldsymbol{A},1)\leqslant p(\boldsymbol{A},2)\leqslant\cdots\leqslant p(\boldsymbol{A},n).$$

证明对 $i=1:n$ 有 $m(\boldsymbol{A},i)=m(\boldsymbol{L},i)$ 和 $p(\boldsymbol{A},i)=p(\boldsymbol{U},i)$. $m(\boldsymbol{A},i)$ 的定义见习题 4.3.4.

**4.3.7** 将算法 4.3.1 改成基于 gaxpy 运算的形式.

**4.3.8** 设计一个整体流, 可向量化的算法来求解对称正定三对角方程组 $\boldsymbol{A}^{(k)}\boldsymbol{x}^{(k)}=\boldsymbol{b}^{(k)}$. 假设对角线元、次对角线元和右端向量按行存储于数组 $\boldsymbol{D},\boldsymbol{E}$ 和 $\boldsymbol{B}$ 中, $\boldsymbol{b}^{(k)}$ 被 $\boldsymbol{x}^{(k)}$ 覆盖.

**4.3.9** 将 $\boldsymbol{A}$ 按对角线存储重新设计算法 4.3.1.

**4.3.10** 给出一个 $3\times 3$ 对称正定矩阵的三对角部分不是对称正定的例子.

**4.3.11** 考虑 $\boldsymbol{Ax}=\boldsymbol{b}$ 问题, 其中

$$\boldsymbol{A}=\begin{bmatrix} 2 & -1 & 0 & \cdots & 0 & -1 \\ -1 & 2 & -1 & \ddots & \vdots & 0 \\ 0 & -1 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \ddots & 2 & -1 \\ -1 & 0 & \cdots & 0 & -1 & 2 \end{bmatrix}.$$

这类矩阵在具有周期性边界条件的边值问题中出现, (a) 证明 $\boldsymbol{A}$ 是奇异的. (b) 给出 $\boldsymbol{b}$ 必须满足的使得该方程组存在解的条件, 并设计一个算法求解. (c) 假设 $n$ 为偶数, 考虑置换矩阵 $\boldsymbol{P}=[e_1\ \ e_n\ \ e_2\ \ e_{n-1}\ \ e_3\cdots]$, 其中 $e_k$ 是 $\boldsymbol{I}_n$ 的第 $k$ 列. 试描述变换了的方程组 $\boldsymbol{P}^{\mathrm T}\boldsymbol{AP}(\boldsymbol{P}^{\mathrm T}\boldsymbol{x})=\boldsymbol{P}^{\mathrm T}\boldsymbol{b}$, 并给出解法 (假定解存在并忽略选主元).

## 本节注释与参考文献

关于带状方程组的文献不计其数, 其中有代表性的包括:

R. S. Martin and J. H. Wilkinson (1965). "Symmetric Decomposition of Positive Definite Band Matrices," *Numer. Math.* 7, 355–361.

R. S. Martin and J. H. Wilkinson (1967). "Solution of Symmetric and Unsymmetric Band Equations and the Calculation of Eigenvalues of Band Matrices," *Numer. Math.* 9, 279–301.

E. L. Allgower (1973). "Exact Inverses of Certain Band Matrices," *Numer. Math.* 21, 279–284.

Z. Bohte (1975). "Bounds for Rounding Errors in the Gaussian Elimination for Band Systems," *J. Inst. Math. Applic.* 16, 133–142.

I. S. Duff (1977). "A. Survey of Sparse Matrix Research," *Proc. IEEE* 65, 500–535.

N. J. Higham (1990). "Bounding the Error in Gaussian Elimination for Tridiagonal Systems," *SIAM J. Matrix Anal. Appl.* 11, 521–530.

关于带状矩阵的研究, 相当多的精力花费在如何减小其带宽, 参阅:

E. Cuthill(1972). "Several Strategies for Reducing the Bandwidth of Matrices," in *Sparse Matrices and Their Applications,* ed. D. J. Rose and R. A. Willoughby, Plenum Press, New York.

N. E. Gibbs, W. G. Poole, Jr., and P. K. Stockmeyer (1976)." An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix," *SIAM J. Num. Anal.* 13, 236–250.

N. E. Gibbs, W. G. Poole, Jr., and P. K. Stockmeyer (1976). "A Comparison of Several Bandwidth and Profile Reduction Algorithms," *ACM Trans. Math. Soft.* 2, 322–330.

我们曾提到, 三对角方程组出现的频率特高, 因此不奇怪有相当的注意力集中在寻求解决此类问题的方法上.

C. Fischer and R. A. Usmani (1969). "Properties of Some Tridiagonal Matrices and Their Application to Boundary Value Problems," *SIAM J. Num. Anal.* 6, 127–142.

D. J. Rose(1969). "An Algorithm for Solving a Special Class of Tridiagonal Systems of Linear Equations," *Comm. ACM* 12, 234–236.

H. S. Stone (1973). "An Efficient Parallel Algorithm for the Solution of a Tridiagonal Linear System of Equations," *J. ACM* 20, 27–38.

M. A. Malcolm and J. Palmer (1974). "A Fast Method for Solving a Class of Tridiagonal Systems of Linear Equations," *Comm. ACM* 17, 14–17.

J. Lambiotte and R. G. Voigt (1975). "The Solution of Tridiagonal Linear Systems of the CDC-STAR 100 Computer," *ACM Trans. Math. Soft.* 1, 308–329.

H. S. Stone (1975). "Parallel Tridiagonal Equation Solvers," *ACM Trans. Math. Soft.* 1, 289–307.

D. Kershaw (1982). "Solution of Single Tridiagonal Linear Systems and Vectorization of the ICCG Algorithm on the Gray-1," in G. Roderigue (ed), *Parallel Computation*, Academic Press, NY, 1982.

N. J. Higham (1986). "Efficient Algorithms for computing the condition number of a tridiagonal matrix," *SIAM J. Sci. and Stat. Comp.* 7, 150–165.

George and Liu (1981) 的第 4 章有一个关于正定矩阵的带状方法的很好的综述.

## 4.4 对称不定方程组

一个对称矩阵, 如果其二次型 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$ 既可取正值又可为负值, 则被称为*不定的*. 尽管不定矩阵也存在 $\mathrm{LDL}^{\mathrm{T}}$ 分解, 但其矩阵因子的元素可能任意大:

$$\begin{bmatrix} \varepsilon & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1/\varepsilon & 1 \end{bmatrix} \begin{bmatrix} \varepsilon & 0 \\ 0 & -1/\varepsilon \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1/\varepsilon & 1 \end{bmatrix}^{\mathrm{T}}.$$

当然, 3.4 节中的选主元法都可被用来解决此问题. 但这会破坏对称性同时也会破坏掉寻找 "Cholesky 速度" 的不定方程组解的机会. 对称选主元方法, 即将数据以 $\boldsymbol{A} \leftarrow \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}}$ 形式重新组织, 必须如 4.2.9 节讨论的那样来使用. 不幸的是, 对称选主元法并不总是能保证 $\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}$ 计算的稳定性. 如果 $\varepsilon_1$ 和 $\varepsilon_2$ 值很小, 则无论什么 $\boldsymbol{P}$, 矩阵

$$\widetilde{\boldsymbol{A}} = \boldsymbol{P} \begin{bmatrix} \varepsilon_1 & 1 \\ 1 & \varepsilon_2 \end{bmatrix} \boldsymbol{P}^{\mathrm{T}}$$

的对角元素依然非常小, 分解总会出现大的数. 对称选主元过程中, 主元总是从对角线上选取的, 如这些数比需要消零的非对角元小得多, 就会引起麻烦. 因此对称选主元的 $\mathrm{LDL}^{\mathrm{T}}$ 分解不能作为求解对称不定方程组的可靠方法. 看来, 选主元时要考虑非对角元同时又要保持对称性是一个挑战.

本节我们讨论两种面临这一挑战的方法. 第一种由 Aasen (1971) 提出, 它进行如下分解:

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{T}\boldsymbol{L}^{\mathrm{T}}, \tag{4.4.1}$$

其中 $\boldsymbol{L} = (l_{ij})$ 是单位下三角形矩阵, $\boldsymbol{T}$ 是三对角矩阵. $\boldsymbol{P}$ 是能使 $|l_{ij}| \leqslant 1$ 的置换矩阵. 与此相对应, Bunch and Parlett (1971) 提出的*对角选主元法*计算置换矩阵 $\boldsymbol{P}$ 使得

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}, \tag{4.4.2}$$

其中 $\boldsymbol{D}$ 是由 $1 \times 1$ 和 $2 \times 2$ 的块构成的块对角矩阵. 同样, $\boldsymbol{P}$ 的选取使得下三角形矩阵 $\boldsymbol{L}$ 的元素满足 $|l_{ij}| \leqslant 1$. 两种分解方法都需要 $n^3/3$ 个 flop. 一旦完成分解则可在 $O(n^2)$ 的工作量内解出 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$:

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{T}\boldsymbol{L}^{\mathrm{T}}, \boldsymbol{L}\boldsymbol{z} = \boldsymbol{P}\boldsymbol{b}, \boldsymbol{T}\omega = z, \boldsymbol{L}^{\mathrm{T}}\boldsymbol{y} = \omega, \boldsymbol{x} = \boldsymbol{P}\boldsymbol{y} \Rightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},$$

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\mathrm{T}} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}, \boldsymbol{L}\boldsymbol{z} = \boldsymbol{P}\boldsymbol{b}, \boldsymbol{D}\omega = \boldsymbol{z}, \boldsymbol{L}^{\mathrm{T}}\boldsymbol{y} = \omega, \boldsymbol{x} = \boldsymbol{P}\boldsymbol{y} \Rightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

在这些求解过程中需要讨论的 "新" 东西是方程组 $\boldsymbol{T}\boldsymbol{w} = \boldsymbol{z}$ 和 $\boldsymbol{D}\boldsymbol{w} = \boldsymbol{z}$.

在 Aasen 方法中, 对称的不定三对角方程组 $\boldsymbol{T}\boldsymbol{w} = \boldsymbol{z}$ 可应用选主元的带状高斯消去法在 $O(n)$ 时间内解出. 请注意, 在此层次上忽略对称性不需付出很大代价, 因为整个开销为 $O(n^3)$.

在对角选主元法中, 方程组 $\boldsymbol{D}\boldsymbol{w}=\boldsymbol{z}$ 相当于一组 $1\times1$ 和 $2\times2$ 对称不定方程组. $2\times2$ 的问题可由选主元的高斯消去法解决. 再次说明的是, 在这 $O(n)$ 次计算中抛弃对称性并没有什么不妥.

因此, 本节的核心问题便是如何有效地计算分解 (4.4.1) 和 (4.4.2).

### 4.4.1 Parlett-Reid 算法

Parlett and Reid(1970) 给出了如何用高斯变换计算 (4.4.1). 我们用 $n=5$ 的矩阵在第 $k=2$ 步的执行情况来说明这个算法. 在这步的开始, 矩阵 $\boldsymbol{A}$ 已被化为

$$\boldsymbol{A}^{(1)}=\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{M}_1^{\mathrm{T}}=\begin{bmatrix}\alpha_1&\beta_1&0&0&0\\\beta_1&\alpha_2&v_3&v_4&v_5\\0&v_3&\times&\times&\times\\0&v_4&\times&\times&\times\\0&v_5&\times&\times&\times\end{bmatrix},$$

其中置换矩阵 $\boldsymbol{P}_1$ 的选取使得高斯变换阵 $\boldsymbol{M}_1$ 的元素不大于 1. 检查 $(v_3,v_4,v_5)^{\mathrm{T}}$ 中的最大元素, 确定一个 $3\times3$ 置换矩阵 $\widetilde{\boldsymbol{P}}_2$, 使得

$$\widetilde{\boldsymbol{P}}_2\begin{bmatrix}v_3\\v_4\\v_5\end{bmatrix}=\begin{bmatrix}\widetilde{v}_3\\\widetilde{v}_4\\\widetilde{v}_5\end{bmatrix}\Rightarrow|\widetilde{v}_3|=\max\{|\widetilde{v}_3|,|\widetilde{v}_4|,|\widetilde{v}_5|\}.$$

如果最大元素为 0, 则令 $\boldsymbol{M}_2=\boldsymbol{P}_2=\boldsymbol{I}$ 并继续下一步. 否则, 令 $\boldsymbol{P}_2=\mathrm{diag}(\boldsymbol{I}_2,\widetilde{\boldsymbol{P}}_2)$, $\boldsymbol{M}_2=\boldsymbol{I}-\alpha^{(2)}\boldsymbol{e}_3^{\mathrm{T}}$, 其中

$$\alpha^{(2)}=(0\ \ 0\ \ 0\ \ \widetilde{v}_4/\widetilde{v}_3\ \ \widetilde{v}_5/\widetilde{v}_3)^{\mathrm{T}},$$

则会有

$$\boldsymbol{A}^{(2)}=\boldsymbol{M}_2\boldsymbol{P}_2\boldsymbol{A}^{(1)}P_2^{\mathrm{T}}\boldsymbol{M}_2^{\mathrm{T}}=\begin{bmatrix}\alpha_1&\beta_1&0&0&0\\\beta_1&\alpha_2&\widetilde{v}_3&0&0\\0&\widetilde{v}_3&\times&\times&\times\\0&0&\times&\times&\times\\0&0&\times&\times&\times\end{bmatrix}.$$

一般地, 算法执行 $n-2$ 步后得到三对角矩阵

$$\boldsymbol{T}=\boldsymbol{A}^{(n-2)}=(\boldsymbol{M}_{n-2}\boldsymbol{P}_{n-2}\cdots\boldsymbol{M}_1\boldsymbol{P}_1)\boldsymbol{A}(\boldsymbol{M}_{n-2}\boldsymbol{P}_{n-2}\cdots\boldsymbol{M}_1\boldsymbol{P}_1)^{\mathrm{T}}.$$

可证, 若有 $\boldsymbol{P}=\boldsymbol{P}_{n-2}\cdots\boldsymbol{P}_1$ 和

$$\boldsymbol{L}=(\boldsymbol{M}_{n-2}\boldsymbol{P}_{n-2}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{P}^{\mathrm{T}})^{-1},$$

则 (4.4.1) 成立. 考察 $\boldsymbol{L}$ 可发现 $\boldsymbol{L}$ 的第 1 列是 $\boldsymbol{e}_1$, 其第 $k$ 列 $(k>1)$ 的次对角线上的元素是由 $\boldsymbol{M}_{k-1}$ 的乘子组成的.

若想有效地实现 Parlett-Reid 方法, 要小心地计算修正矩阵

$$A^{(k)} = M_k(P_k A^{(k-1)} P_k^{\mathrm{T}}) M_k^{\mathrm{T}}. \tag{4.4.3}$$

为了尽量少用记号来说明所涉及的问题, 假定 $B = B^{\mathrm{T}}$ 是 $n-k$ 阶的而且想计算 $B_+ = (I - \omega e_1^{\mathrm{T}}) B (I - \omega e_1^{\mathrm{T}})^{\mathrm{T}}$, 其中 $\omega \in \mathbb{R}^{n-k}$, $e_1$ 是 $I_{n-k}$ 的第 1 列. 此计算是 (4.4.3) 的核心, 如令

$$u = Be_1 - \frac{b_{11}}{2}\omega,$$

则对称矩阵 $B_+ = B - \omega u^{\mathrm{T}} - u\omega^{\mathrm{T}}$ 的下半部可在 $2(n-k)^2$ 个 flop 内算出. 从 $k=1$ 到 $k=n-2$ 累加此量, 即知 Parlett-Reid 算法共需 $2n^3/3$ 个 flop, 这是我们所期望的两倍.

**例 4.4.1** 如果 Parlett-Reid 算法应用于

$$A = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \\ 3 & 2 & 3 & 4 \end{bmatrix},$$

则

$$\begin{aligned} P_1 &= [e_1 \;\; e_4 \;\; e_3 \;\; e_2], \\ M_1 &= I_4 - (0,0,2/3,1/3)^{\mathrm{T}} e_2^{\mathrm{T}}, \\ P_2 &= [e_1 \;\; e_2 \;\; e_4 \;\; e_3], \\ M_2 &= I_4 - (0,0,0,1/2)^{\mathrm{T}} e_3^{\mathrm{T}}, \end{aligned}$$

且 $PAP^{\mathrm{T}} = LTL^{\mathrm{T}}$, 其中 $P = [e_1 \;\; e_3 \;\; e_4 \;\; e_2]$,

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1/3 & 1 & 0 \\ 0 & 2/3 & 1/2 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} 0 & 3 & 0 & 0 \\ 3 & 4 & 2/3 & 0 \\ 0 & 2/3 & 10/9 & 0 \\ 0 & 0 & 0 & 1/2 \end{bmatrix}.$$

### 4.4.2 Aasen 方法

通过重新考虑 Parlett-Reid 算法中某些计算过程, Aasen (1971) 提出一种能在 $n^3/3$ 个 flop 内计算 (4.4.1) 的方法. 我们需要三对角矩阵的记号:

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{n-1} \\ 0 & \cdots & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

为清晰起见, 暂时不考虑选主元, 假定分解 $A = LTL^{\mathrm{T}}$ 存在, 其中 $L$ 是单位下三角形矩阵且 $L(:,1) = e_1$. Aasen 方法是如下进行的:

$$
\begin{array}{l}
\textbf{for } j=1:n \\
\quad \text{计算 } h(1:j), \text{ 这里 } h=TL^{\mathrm{T}}e_j=He_j \\
\quad \text{计算 } \alpha(j) \\
\quad \textbf{if } j\leqslant n-1 \\
\quad\quad \text{计算 } \beta(j) \\
\quad \textbf{end} \\
\quad \textbf{if } j\leqslant n-2 \\
\quad\quad \text{计算 } L(j+2:n,j+1) \\
\quad \textbf{end} \\
\textbf{end}
\end{array}
\tag{4.4.4}
$$

因此 Aasen 算法的第 $j$ 步的任务是计算 $\boldsymbol{T}$ 的第 $j$ 列和 $\boldsymbol{L}$ 的第 $(j+1)$ 列. 这算法利用了 $\boldsymbol{H}=\boldsymbol{TL}^{\mathrm{T}}$ 是上 Hessenberg 矩阵这一事实. 由 (4.4.4) 可推知 $\alpha(j),\beta(j)$ 和 $L(j+2:n,j+1)$ 的计算由向量 $h(1:j)=H(1:j,j)$ 决定, 让我们看一看其中的原因.

考虑方程 $\boldsymbol{A}=\boldsymbol{LH}$ 的第 $j$ 列:

$$
A(:,j)=L(:,1:j+1)h(1:j+1). \tag{4.4.5}
$$

这说明 $A(i,j)$ 是 $\boldsymbol{L}$ 的前 $j+1$ 列的线性组合. 特别地,

$$
A(j+1:n,j)=L(j+1:n,1:j)h(1:j)+L(j+1:n,j+1)h(j+1).
$$

如果我们计算

$$
v(j+1:n)=A(j+1:n,j)-L(j+1:n,1:j)h(1:j),
$$

则

$$
L(j+1:n,j+1)h(j+1)=v(j+1:n). \tag{4.4.6}
$$

因此, $L(i+2:n,j+1)$ 是 $v(j+2:n)$ 的倍数 (scaling). 因为 $L$ 是单位下三角形矩阵, 由 (4.4.6) 得

$$
v(j+1)=h(j+1),
$$

于是从同一方程可得到求 $\boldsymbol{L}$ 的第 $(j+1)$ 列的方法,

$$
L(j+2:n,j+1)=v(j+2:n)/v(j+1).
$$

请注意 $L(j+2:n,j+1)$ 是加权 gaxpy 运算.

接下来给出求 $\alpha(j)$ 和 $\beta(j)$ 的公式. 比较方程 $\boldsymbol{H}=\boldsymbol{TL}^{\mathrm{T}}$ 中的 $(j,j)$ 元和 $(j+1,j)$ 元. 利用约定 $\beta(0)=0$, 我们发现 $h(j)=\beta(j-1)L(j,j-1)+\alpha(j)$ 和 $h(j+1)=v(j+1)$, 于是

$$
\begin{aligned}
\alpha(j)&=h(j)-\beta(j-1)L(j,j-1),\\
\beta(j)&=v(j+1).
\end{aligned}
$$

由上我们可以写出完整的 Aasen 算法:

$$
\begin{array}{l}
\textbf{for } j=1:n \\
\quad \text{计算 } h(1:j), \text{ 这里 } h=TL^{\mathrm{T}}e_j \\
\quad \textbf{if } j=1 \vee j=2 \\
\qquad \alpha(j)=h(j) \\
\quad \textbf{else} \\
\qquad \alpha(j)=h(j)-\beta(j-1)L(j,j-1) \\
\quad \textbf{end} \\
\quad \textbf{if } j\leqslant n-1 \\
\qquad v(j+1:n)=A(j+1:n,j)-L(j+1:n,1:j)h(1:j) \\
\qquad \beta(j)=v(j+1) \\
\quad \textbf{end} \\
\quad \textbf{if } j\leqslant n-2 \\
\qquad L(j+2:n,j+1)=v(j+2:n)/v(j+1) \\
\quad \textbf{end} \\
\textbf{end}
\end{array} \tag{4.4.7}
$$

我们还需详述 $h(1:j)$ 的计算过程. 由 (4.4.5) 得

$$A(1:j,j)=L(1:j,1:j)h(1:j). \tag{4.4.8}$$

由于已知 $\boldsymbol{L}$ 的前 $j$ 列, 故这个下三角方程组可用来求解 $h(1:j)$. 然而利用方程 $\boldsymbol{H}=\boldsymbol{T}\boldsymbol{L}^{\mathrm{T}}$ 的第 $j$ 列可以得到求解 $H(1:j,j)$ 的更有效的方法. 确切地说, 约定 $\beta(0)L(j,0)=0$, 则对于 $k=1:j$, 我们有

$$h(k)=\beta(k-1)L(j,k-1)+\alpha(k)L(j,k)+\beta(k)L(j,k+1).$$

除了 $k=j$ 的情形之外, 这公式可直接用来计算, 因为我们还没计算出 $\alpha(j)$ 和 $\beta(j)$. 然而一旦 $h(1:j-1)$ 已知, 可由三角方程组 (4.4.8) 最后一行求得 $h(j)$, 即

$$h(j)=A(j,j)-\sum_{k=1}^{j-1}L(j,k)h(k).$$

综合上面结论, 用数组 $l(1:n)$ 来存储 $L(j,1:j)$, (4.4.7) 中的 $h(1:j)$ 的求解过程如下:

$$
\begin{array}{l}
\textbf{if } j=1 \\
\quad h(1)=A(1,1) \\
\textbf{elseif } j=2 \\
\quad h(1)=\beta(1); h(2)=A(2,2) \\
\textbf{else} \\
\quad l(0)=0; l(1)=0; l(2:j-1)=L(j,2:j-1); l(j)=1
\end{array} \tag{4.4.9}
$$

```
        h(j) = A(j,j)
        for k = 1 : j-1
            h(k) = β(k-1)l(k-1) + α(k)l(k) + β(k)l(k+1)
            h(j) = h(j) - l(k)h(k)
        end
    end
```

注意, 用这个$O(j)$的方法来计算$h(1:j)$, 则算法 (4.4.7) 中的主要工作是求 $v(j+1:n)$ 的 gaxpy 运算. 在第 $j$ 步中, gaxpy 需 $2j(n-j)$ 个 flop. 累加 $j=1:n$ 可知 Aasen 算法需要 $n^3/3$ 个 flop. 因此 Aasen 算法和 Cholesky 算法的运算量是相同的.

### 4.4.3 选主元的 Aasen 方法

现在我们知道, $\boldsymbol{L}$ 的列是由 (4.4.7) 中的 $v$ 向量做数乘得到的. 如果那个数乘因子很大, 也就是说某个 $v(j+1)$ 很小, 就会引起麻烦. 为避免出现这个问题, 只需将 $v(j+1:n)$ 中的最大元素置换到最上面. 当然, 此置换变换必须恰当地作用于 $\boldsymbol{A}$ 的未约化部分和 $\boldsymbol{L}$ 的已计算部分。

**算法 4.4.1 (Aasen 方法)** 如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称的, 则本算法计算出一个置换矩阵 $\boldsymbol{P}$, 一个单位下三角形矩阵 $\boldsymbol{L}$ 和一个三对角形矩阵 $\boldsymbol{T}$, 使得 $\boldsymbol{PAP}=\boldsymbol{LTL}^{\mathrm{T}}$, 且 $|L(i,j)|\leqslant 1$, 置换矩阵 $P$ 由整数向量 piv 来标记. 确切地说, $\boldsymbol{P}=\boldsymbol{P}_1\cdots\boldsymbol{P}_{n-2}$, 其中 $\boldsymbol{P}_j$ 是将单位矩阵的第 piv(j) 行和第 $j+1$ 行互换所得. $\boldsymbol{T}$ 的对角元和次对角元分别存储于 $\alpha(1:n)$ 和 $\beta(1:n-1)$ 中, 只计算 $L(2:n,2:n)$ 的次对角线部分:

```
for j = 1 : n
    由 (4.4.9) 计算 h(1:j)
    if j = 1 ∨ j = 2
        α(j) = h(j)
    else
        α(j) = h(j) - β(j-1)L(j,j-1)
    end
    if j ⩽ n-1
        v(j+1:n) = A(j+1:n,j) - L(j+1:n,1:j)h(1:j)
        找 q 使得 |v(q)| = ||v(j+1:n)||∞, 这里 j+1 ⩽ q ⩽ n
        piv(j) = q; v(j+1) ↔ v(q); L(j+1,2:j) ↔ L(q,2:j)
        A(j+1,j+1:n) ↔ A(q,j+1:n)
        A(j+1:n,j+1) ↔ A(j+1:n,q)
        β(j) = v(j+1)
    end
    if j ⩽ n-2
```

```
            L(j+2:n, j+1) = v(j+2:n)
            if v(j+1) ≠ 0
                L(j+2:n, j+1) = L(j+2:n, j+1)/v(j+1)
            end
        end
    end
```

Aasen 方法与列选主高斯消去法在同样意义上是稳定的. 这就是说, 可以得到一个靠近 $\boldsymbol{A}$ 的矩阵之精确分解, 只要 $\|\hat{\boldsymbol{T}}\|_2/\|\boldsymbol{A}\|_2 \approx 1$, 其中 $\hat{\boldsymbol{T}}$ 是三对角矩阵 $\boldsymbol{T}$ 的计算值. 通常, 这个结论几乎总是对的.

在 Aasen 算法的实际执行过程中, 用 $\boldsymbol{L}$ 和 $\boldsymbol{T}$ 来覆盖 $\boldsymbol{A}$ 的下三角部分. 下面是 $n=5$ 的情形:

$$\boldsymbol{A} \leftarrow \begin{bmatrix} \alpha_1 & & & & \\ \beta_1 & \alpha_2 & & & \\ l_{32} & \beta_2 & \alpha_3 & & \\ l_{42} & l_{43} & \beta_3 & \alpha_4 & \\ l_{52} & l_{53} & l_{54} & \beta_4 & \alpha_5 \end{bmatrix}$$

注意在这个排列中, $\boldsymbol{L}$ 的列均向左移了.

### 4.4.4 对角选主元法

下面讨论分块 LDL$^{\mathrm{T}}$ 分解 (4.4.2) 的计算. 我们仿效 Bunch and Parlett(1971) 的讨论. 假定

$$\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{P}_1^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{E} & \boldsymbol{C}^{\mathrm{T}} \\ \boldsymbol{C} & \boldsymbol{B} \end{bmatrix} \begin{matrix} s \\ n-s \end{matrix}$$
$$\qquad\qquad\quad s \quad n-s$$

其中 $\boldsymbol{P}_1$ 是置换矩阵且 $s=1$ 或 $s=2$. 如果 $\boldsymbol{A}$ 非零, 则总可以挑选这些量使 $\boldsymbol{E}$ 非奇异, 从而可写成

$$\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{P}_1^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{I}_s & 0 \\ \boldsymbol{C}\boldsymbol{E}^{-1} & \boldsymbol{I}_{n-s} \end{bmatrix} \begin{bmatrix} \boldsymbol{E} & 0 \\ 0 & \boldsymbol{B}-\boldsymbol{C}\boldsymbol{E}^{-1}\boldsymbol{C}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_s & \boldsymbol{E}^{-1}\boldsymbol{C}^{\mathrm{T}} \\ 0 & \boldsymbol{I}_{n-s} \end{bmatrix}.$$

考虑到稳定性, $s\times s$ 的 "主元" $\boldsymbol{E}$ 的选取应使得

$$\widetilde{\boldsymbol{A}} = (\widetilde{a}_{ij}) \equiv \boldsymbol{B} - \boldsymbol{C}\boldsymbol{E}^{-1}\boldsymbol{C}^{\mathrm{T}} \tag{4.4.10}$$

中的元素 $a_{ij}$ 被适当限界. 为此, 设 $\alpha \in (0,1)$ 已给定, 定义度量:

$$\mu_0 = \max_{i,j} |a_{ij}|, \quad \mu_1 = \max_i |a_{ii}|.$$

Bunch-Parlett 的选主元方案如下:

```
    if μ1 ⩾ αμ0
        s = 1
```

　　　　选取 $P_1$ 使得 $|e_{11}| = \mu_1$

　　else

　　　　$s = 2$

　　　　选取 $P_1$ 使得 $|e_{21}| = \mu_0$

　　end

由 (4.4.10) 容易证得, 如果 $s = 1$, 则

$$|\tilde{a}_{ij}| \leqslant (1 + \alpha^{-1})\mu_0. \tag{4.4.11}$$

若 $s = 2$, 则

$$|\widetilde{a}_{ij}| \leqslant \frac{3 - \alpha}{1 - \alpha}\mu_0. \tag{4.4.12}$$

令 $(1 + \alpha^{-1})^2$ 与 $(3 - \alpha)/(1 - \alpha)$ 相等, 其中 $(1 - \alpha^{-1})^2$ 是对应于两步 $s = 1$ 的增长因子, $(3 - \alpha)/(1 - \alpha)$ 是对应于 $s = 2$ 的增长因子, Bunch 和 Parlett 得出 $\alpha = (1 + \sqrt{17})/8$ 是在极小化元素增长界的意义下之最佳值.

然后, 将上面给出的约化过程用到 $n - s$ 阶对称矩阵 $\widetilde{\boldsymbol{A}}$. 利用简单的归纳法可知, 分解 (4.4.2) 存在, 如果不考虑选主元的工作量, 则分解运算需 $n^3/3$ 个 flop.

### 4.4.5 稳定性及效率

Bunch(1971) 证明了上述对角选主元法是与全选主元的高斯消去法同样稳定的. 不幸的是, 整个过程却需做 $n^3/12 \sim n^3/6$ 次比较运算, 因为在每步约化 $\mu_0$ 都涉及二维搜索. 实际的比较次数依赖于 $2 \times 2$ 矩阵主元的数目, 但一般情况下计算 (4.4.2) 的 Bunch-Parlett 方法要比 Aasen 法慢得多. 参见 Barwell and George(1976).

Barwell and Kaufman(1977) 的对角选主元方法却不存在这个问题. 他们的算法中每步约化仅需搜索两列. 考虑约化的第一步便可充分说明这一技巧.

　　$\alpha = (1 + \sqrt{17})/8$; $\lambda = |a_{r1}| = \max\{|a_{21}|, \cdots, |a_{n1}|\}$

　　**if** $\lambda > 0$

　　　　**if** $|a_{11}| \geqslant \alpha\lambda$

　　　　　　$s = 1; P_1 = I$

　　　　**else**

　　　　　　$\sigma = |a_{pr}| = \max\{|a_{1r}, \cdots, |a_{r-1,r}|, |a_{r+1,r}|, \cdots, |a_{nr}|\}$

　　　　　　**if** $\sigma|a_{11}| \geqslant \alpha\lambda^2$

　　　　　　　　$s = 1, P_1 = I$

　　　　　　**elseif** $|a_{rr}| \geqslant \alpha\sigma$

　　　　　　　　$s = 1$, 选取 $P_1$ 使得 $(P_1^{\mathrm{T}}AP_1)_{11} = a_{rr}$

　　　　　　**else**

　　　　　　　　$s = 2$, 选取 $P_1$ 使得 $(P_1^{\mathrm{T}}AP_1)_{21} = a_{rp}$

　　　　　　**end**

```
        end
    end
```

Bunch-Kaufman 算法总体上需要 $n^3/3$ 个 flop, $O(n^2)$ 次比较, 而且和本节其他算法一样需要 $n^2/2$ 个存储单元.

**例 4.4.2** 如果将 Bunch-Kaufman 算法应用于

$$A = \begin{bmatrix} 1 & 10 & 20 \\ 10 & 1 & 30 \\ 20 & 30 & 1 \end{bmatrix},$$

则第一步有 $\lambda = 20, r = 3, \sigma = 30, p = 2$. 置换矩阵 $P = [e_3 \ \ e_2 \ \ e_1]$ 的作用结果为

$$PAP^{\mathrm{T}} = \begin{bmatrix} 1 & 30 & 20 \\ 30 & 1 & 10 \\ 20 & 10 & 1 \end{bmatrix}.$$

然后一个 $2 \times 2$ 的主元用来产生如下约化

$$PAP^{\mathrm{T}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.3115 & 0.6563 & 1 \end{bmatrix} \begin{bmatrix} 1 & 30 & 0 \\ 30 & 1 & 0 \\ 0 & 0 & -11.7920 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.3115 & 0.6563 & 1 \end{bmatrix}^{\mathrm{T}}.$$

### 4.4.6 关于平衡方程组的说明

有一类重要的对称非定矩阵形如

$$A = \begin{bmatrix} C & B \\ B^{\mathrm{T}} & 0 \end{bmatrix} \begin{matrix} n \\ p \end{matrix} \qquad (4.4.13)$$
$$\qquad\quad n \qquad p$$

其中 $C$ 是对称正定矩阵, $B$ 是列满秩的. 这两个条件保证了 $A$ 是非奇异的.

当然, 本节的那些方法同样适用于 $A$. 但是这些方法没有利用上这类矩阵的特殊结构, 因为选主元方法破坏了 (2,2) 位置上的 0 块, 下面的方法尝试利用 $A$ 的分块结构特点.

(a) 计算 $C$ 的 Cholesky 分解 $C = GG^{\mathrm{T}}$;

(b) 从 $GK = B$ 中解 $K \in \mathbb{R}^{n \times p}$;

(c) 计算 $K^{\mathrm{T}}K = B^{\mathrm{T}}C^{-1}B$ 的 Cholesky 分解 $HH^{\mathrm{T}} = K^{\mathrm{T}}K$.

于是

$$A = \begin{bmatrix} G & 0 \\ K^{\mathrm{T}} & H \end{bmatrix} \begin{bmatrix} G^{\mathrm{T}} & K \\ 0 & -H^{\mathrm{T}} \end{bmatrix}$$

理论上, 这种三角分解可用来求解平衡方程组:

$$\begin{bmatrix} \boldsymbol{C} & \boldsymbol{B} \\ \boldsymbol{B}^{\mathrm{T}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}. \tag{4.4.14}$$

然而, 考察 (b) 步和 (c) 步可清楚看到计算解的准确度依赖于 $\kappa(\boldsymbol{C})$, 且其值可能比 $\kappa(\boldsymbol{A})$ 大得多. 人们已对这一情形进行了仔细的分析并提出了许多利用结构的算法. 在本节的最后简要回顾了此方面的文献.

在结束本节之前考虑 (4.4.14) 的一个特例, 此例清楚地指出, 一个算法是稳定的意味着什么和扰动分析如何能够引导人们寻找更好的方法. 在许多重要的应用场合 $g=0, \boldsymbol{C}$ 是对角矩阵, 且解的子向量 $y$ 是非常重要的. 对 (4.4.14) 做变换, $y$ 可表示为

$$y = (\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}f. \tag{4.4.15}$$

此式再一次让我们觉得 $\kappa(\boldsymbol{C})$ 对计算出的 $y$ 值的准确度有影响. 但是, 可证明

$$\|(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}\| \leqslant \psi_B, \tag{4.4.16}$$

其中上界 $\psi_B$ 是与 $\boldsymbol{C}$ 无关的, 这表明 $y$ 对于 $\boldsymbol{C}$ 中的扰动是不灵敏的. 求解此向量的稳定方法应遵循上述原则, 即计算得的 $y$ 值的准确度应是不依赖于 $\boldsymbol{C}$ 的. Vavasis(1994) 给出了一个具有该性质的方法. 它涉及仔细地形成一个矩阵 $\boldsymbol{V} \in \mathbb{R}^{n\times(n-p)}$, 其列向量是 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}$ 之零空间的一组基. 然后, 求解 $n\times n$ 线性方程组

$$[\boldsymbol{B} \quad \boldsymbol{V}] \begin{bmatrix} y \\ q \end{bmatrix} = f,$$

这保证 $f = \boldsymbol{B}y + \boldsymbol{V}q$. 于是 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{C}^{-1}f = \boldsymbol{B}^{\mathrm{T}}C^{-1}\boldsymbol{B}y$, 故知 (4.4.15) 成立.

## 习 题

**4.4.1** 证明: 如果一个 $n\times n$ 对称矩阵 $\boldsymbol{A}$ 的 $1\times 1$ 和 $2\times 2$ 阶主子矩阵都是奇异的, 则 $\boldsymbol{A}$ 为零矩阵.

**4.4.2** 证明: 如果 $\boldsymbol{A}$ 是正定的, 则 Bunch-Kaufman 算法中不会出现 $2\times 2$ 主元.

**4.4.3** 重新设计算法 4.4.1 使得计算过程中涉及 $\boldsymbol{A}$ 的下三角部分, 对 $j=1:n$ 以 $\alpha(j)$ 覆盖 $A(j,j)$, 对 $j=1:n-1$ 以 $\beta(j)$ 覆盖 $A(j+1,j)$, 对 $j=2:n-1$ 和 $i=j+1:n$, 以 $L(i,j)$ 覆盖 $A(i,j-1)$.

**4.4.4** 假定 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是非奇异的、对称的和严格对角占优的, 写一个算法计算下面分解:

$$\boldsymbol{\Pi}\boldsymbol{A}\boldsymbol{\Pi}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{S} & -\boldsymbol{M} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}^{\mathrm{T}} & \boldsymbol{S}^{\mathrm{T}} \\ \boldsymbol{0} & \boldsymbol{M}^{\mathrm{T}} \end{bmatrix},$$

其中 $\boldsymbol{R}\in\mathbb{R}^{k\times k}$ 和 $\boldsymbol{M}\in\mathbb{R}^{(n-k)\times(n-k)}$ 为非奇异的下三角形矩阵, $\boldsymbol{\Pi}$ 是置换矩阵.

**4.4.5** 证明: 如果

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & -\boldsymbol{A}_{22} \end{bmatrix} \begin{matrix} n \\ p \end{matrix}$$
$$\qquad\quad \begin{matrix} n & \quad p \end{matrix}$$

是对称的并且 $A_{11}$ 和 $A_{22}$ 是正定的, 则 $A$ 有一个 $\mathrm{LDL}^{\mathrm{T}}$ 分解, 且有

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & -D_2 \end{bmatrix},$$

其中 $D_1 \in \mathbb{R}^{n\times n}$ 和 $D_2 \in \mathbb{R}^{p\times p}$ 的对角元都为正.

**4.4.6** 证明 (4.4.11) 和 (4.4.12).

**4.4.7** 证明 $-(B^{\mathrm{T}}C^{-1}B)^{-1}$ 是 $A^{-1}$ 的 (2,2) 块, 其中 $A$ 由 (4.4.13) 给出.

**4.4.8** 本题考虑 (4.4.15) 的特殊情形, 定义矩阵

$$M(\alpha) = (B^{\mathrm{T}}C^{-1}B)^{-1}B^{\mathrm{T}}C^{-1},$$

其中 $C = (I_n + \alpha e_k e_k^{\mathrm{T}})$, $\alpha > -1$, 并且 $e_k = I_n(:,k)$ (注意 $C$ 是将 $\alpha$ 加到单位矩阵的第 $(k,k)$ 个元素后所得的矩阵). 假设 $B \in \mathbb{R}^{n\times p}$ 的秩为 $p$, 证明

$$M(\alpha) = (B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}\left(I_n - \frac{\alpha}{1+\alpha w^{\mathrm{T}}w} e_k w^{\mathrm{T}}\right),$$

其中 $w = (I_n - B(B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}})e_k$. 证明: 如果 $\|w\|_2 = 0$ 或 $\|w\|_2 = 1$, 则 $\|M(\alpha)\|_2 = 1/\sigma_{\min}(B)$, 如果 $0 < \|w\|_2 < 1$, 则

$$\|M(\alpha)\|_2 \leqslant \max\left\{\frac{1}{1-\|w\|_2}, 1 + \frac{1}{\|w\|_2}\right\} \Big/ \sigma_{\min}(B).$$

从而 $\|M(\alpha)\|_2$ 有着与 $\alpha$ 无关的上界.

## 本节注释与参考文献

计算 (4.4.1) 的基本参考文献为:

J. O. Aasen (1971). "On the Reduction of a Symmetric Matrix to Tridiagonal Form," *BIT*, 11, 233–242.

B. N. Parlett and J. K. Reid (1970). "On the Solution of a System of Linear Equations Whose Matrix is Symmetric but not Definite," *BIT* 10, 386–397.

对角选主元方面的文献包括:

J. R. Bunch and B. N. Parlett (1971). "Direct Methods for Solving Symmetric Indefinite Systems of Linear Equations," *SIAM J. Num. Anal.* 8, 639–655.

J. R. Bunch (1971). "Analysis of the Diagonal Pivoting Method," *SIAM J. Num. Anal.* 8, 656–680.

J. R. Bunch (1974). "Partial Pivoting Strategies for Symmetric Matrices," *SIAM J. Num. Anal.* 11, 521–528.

J. R. Bunch, L. Kaufman, and B. N. Parlett (1976). "Decomposition of a Symmetric Matrix," *Numer. Math.* 27, 95–109.

J. R. Bunch and L. Kaufman (1977). "Some Stable Methods for Calculating Inertia and Solving Symmetric Linear Systems," *Math. Comp.* 31, 162–179.

I. S. Duff, N. I. M. Gould, J. K. Reid, J. A. Scott, and K. Turner (1991). "The Factorization of Sparse Indefinite Matrices," *IMA J. Num. Anal.* 11, 181–204.

M. T. Jones and M. L. Patrick (1993). "Bunch-Kaufman Factorization for Real Symmetric Indefinite Banded Matrices," *SIAM J. Matrix Anal. Appl.* 14, 553–559.

由于在选主元的过程中必须对还未处理的列进行扫描, 因此要获得一个含有大量 gaxpy 运算的对角选主元算法很困难 (虽然可能). 另一方面, Aasen 方法很自然地含有丰富的 gaxpy 运算. 这两种方法都有分块形式. LAPACK 利用了对角选主元法. 不同的效率在下述文献中有讨论:

V. Barwell and J. A. George (1976). "A Comparison of Algorithms for Solving Symmetric Indefinite Systems of Linear Equations," *ACM Trans. Math. Soft.* 2, 242–251.

M. T. Jones and M. L. Patrick (1994). "Factoring Symmetric Indefinite Matrices on High-Performance Architectures," *SIAM J. Matrix Anal. Appl.* 15, 273–283.

另一种经济的选主元策略利用了基于一种更不严格的交换法则的误差界, 这种思想是从稀疏矩阵的消去法中借用的, 参阅:

R. Fletcher (1976). "Factorizing Symmetric Indefinite Matrices," *Lin. Alg. and Its Applic.* 14, 257–272.

在求解对称方程组 $\boldsymbol{Ax} = \boldsymbol{b}$ 之前, 应当对 $\boldsymbol{A}$ 进行平衡处理. 一个实现这个任务的 $O(n^2)$ 的算法在下述文献中给出:

J. R. Bunch (1971). "Equilibration of Symmetric Matrices in the Max-Norn," *J. ACM* 18, 566–572.

与对称不定方程组类似, 存在求解反对称方程组的解法, 请参阅:

J. R. Bunch (1982). "A Note on the Stable Decomposition of Skew Symmetric Matrices," *Math. Comp.* 158, 475–480.

关于平衡方程组的文献分散于它在其中起重要作用的几个应用领域. 指明这些文献出处的很好的综述文章包括:

G. Strang (1988). "A Framework for Equilibrium Equations," *SIAM Review* 30, 283–297.

S. A. Vavasis (1994). "Stable Numerical Algorithms for Equilibrium Systems," *SIAM J. Matrix Anal. Appl.* 15, 1108–1131.

其他文章包括:

C. C. Paige (1979). "Fast Numerically Stable Computations for Generalized Linear Least Squares Problems," *SIAM J. Num. Anal.* 16, 165–171.

Å. Björck and I. S. Duff (1980). "A Direct Method for the Solution of Sparse Linear Least Squares Problems," *Lin. Alg. and Its Applic.* 34, 43–67.

Å. Björck (1992). "Pivoting and Stability in the Augmented System Method," *Proceedings of the 14th Dundee Conference*, D. F. Griffiths and G. A. Watson (eds), Longman Scientific and Technical, Essex, U. K.

P. D. Hough and S. A. Vavasis (1996). "Complete Orthogonal Decomposition for Weighted Least Squares," *SIAM J. Matrix Anal. Appl.*, to appear.

这些文章中的有些应用了下一章和 12.1 节将讨论的 QR 分解和最小二乘思想.

结构中富含矩阵运算和扰动理论的问题, 在寻求稳定有效的算法方面担当重要角色. 对平衡方程组有类似于 (4.4.15) 的几个重要结果, 它们为最有效的算法奠基. 参阅:

A. Forsgren (1995). "On Linear Least-Squares Problems with Diagonally Dominant Weight Matrices," Technical Report TRITA-MAT-1995-OS2, Department of Mathematics, Royal Institute of Technology, S-100 44, Stockholm, Sweden.

和其中所给出的文献. 对 (4.4.15) 的讨论可在下文中找到:

G. W. Stewart (1989). "On Scaled Projections and Pseudoinverses," *Lin. Alg. and Its Applic.* 112, 189–193.

D. P. O'Leary (1990). "On Bounds for Scaled Projections and Pseudoinverses," *Lin. Alg. and Its Applic.* 132, 115–117.

M. J. Todd (1990). "A Dantzig-Wolfe-like Variant of Karmarker's Interior-Point Linear Programming Algorithm," *Operations Research* 38, 1006–1018.

## 4.5 分块方程组

在许多应用中出现的矩阵都具有可利用的分块结构. 作为一个例子, 我们讨论形如

$$\begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{F}_1 & & \cdots & \boldsymbol{0} \\ \boldsymbol{E}_1 & \boldsymbol{D}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{F}_{n-1} \\ \boldsymbol{0} & \cdots & & \boldsymbol{E}_{n-1} & \boldsymbol{D}_n \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \tag{4.5.1}$$

的分块三对角方程组. 在此我们假定所有的分块都是 $q \times q$ 的, $\boldsymbol{x}_i$ 和 $\boldsymbol{b}_i$ 都属于 $\mathbb{R}^q$. 本节我们讨论求解此类问题的分块 LU 方法和称作循环约化的分而治之方法, 同时简单介绍 Kronecker 积方程组.

### 4.5.1 分块三对角 LU 分解

先来考虑 (4.5.1) 中矩阵的分块 LU 分解. 定义分块三对角矩阵 $\boldsymbol{A}_k$ 为

$$\boldsymbol{A}_k = \begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{F}_1 & & \cdots & \boldsymbol{0} \\ \boldsymbol{E}_1 & \boldsymbol{D}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{F}_{k-1} \\ \boldsymbol{0} & \cdots & & \boldsymbol{E}_{k-1} & \boldsymbol{D}_k \end{bmatrix}, \quad k = 1:n. \tag{4.5.2}$$

比较

$$\boldsymbol{A}_n = \begin{bmatrix} \boldsymbol{I} & & & \cdots & \boldsymbol{0} \\ \boldsymbol{L}_1 & \boldsymbol{I} & & & \vdots \\ & \ddots & \ddots & & \\ \vdots & & \ddots & & \\ \boldsymbol{0} & \cdots & & \boldsymbol{L}_{n-1} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_1 & \boldsymbol{F}_1 & & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{F}_{n-1} \\ \boldsymbol{0} & \cdots & & \boldsymbol{0} & \boldsymbol{U}_n \end{bmatrix} \tag{4.5.3}$$

中的阵, 可得求 $\boldsymbol{L}_i$ 和 $\boldsymbol{U}_i$ 的算法如下:

$U_1 = D_1$
**for** $i = 2:n$
  从 $L_{i-1}U_{i-1} = E_{i-1}$ 解出 $L_{i-1}$ (4.5.4)
  $U_i = D_i - L_{i-1}F_{i-1}$
**end**

只要 $\boldsymbol{U}_i$ 是非奇异的, 这个过程就有定义. 例如, 当矩阵 $\boldsymbol{A}_1, \cdots, \boldsymbol{A}_n$ 是非奇异的时, 这就能够得到保证.

一旦算出 (4.5.3) 的分解, (4.5.1) 中的向量 $\boldsymbol{x}$ 可通过向前和向后迭代来求得:

$y_1 = b_1$
**for** $i = 2:n$
  $y_i = b_i - L_{i-1}y_{i-1}$
**end** (4.5.5)
从 $U_n x_n = y_n$ 解出 $x_n$
**for** $i = n-1:-1:1$
  从 $U_i x_i = y_i - F_i x_{i+1}$ 解出 $x_i$
**end**

为执行 (4.5.4) 和 (4.5.5), 每一个 $\boldsymbol{U}_i$ 必须分解, 因为需要求解以这些子矩阵为系数的线性方程组. 这可通过选主元的高斯消去法来实现. 然而这却不能保证整个算法的稳定性. 只需考虑分块大小 $q$ 为 1 的情况便可看出这点.

### 4.5.2 分块对角占优方程组

为得到令人满意的 $\boldsymbol{L}_i$ 和 $\boldsymbol{U}_i$ 的界, 有必要对分块矩阵做一些假设. 例如, 假设对 $i = 1:n$ 有分块对角占优的关系:

$$\|\boldsymbol{D}_i^{-1}\|_1(\|\boldsymbol{F}_{i-1}\|_1 + \|\boldsymbol{E}_i\|_1) < 1, \quad \boldsymbol{E}_n \equiv \boldsymbol{F}_0 \equiv 0, \tag{4.5.6}$$

那么存在 (4.5.3) 的分解, 且可以证明 $\boldsymbol{L}_i$ 和 $\boldsymbol{U}_i$ 满足不等式

$$\|\boldsymbol{L}_i\|_1 \leqslant 1, \tag{4.5.7}$$

$$\|\boldsymbol{U}_i\|_1 \leqslant \|\boldsymbol{A}_n\|_1. \tag{4.5.8}$$

### 4.5.3 分块解法与带状解法的比较

到此我们很容易提出这样的问题, 为什么不将 (4.5.1) 中的矩阵 $\boldsymbol{A}$ 简单地看作是一个带宽为 $2q-1$ 的以标量为元素的 $qn \times qn$ 矩阵呢? 可应用 4.3 节中给出的带状高斯消去法解之. 这样求解的效率依赖于分块的维数和每块的稀疏模式.

我们通过一个很简单的情形来阐明这一点, 假定需求解

$$\begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{F}_1 \\ \boldsymbol{E}_1 & \boldsymbol{D}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}, \tag{4.5.9}$$

其中 $\boldsymbol{D}_1$ 和 $\boldsymbol{D}_2$ 是对角矩阵, $\boldsymbol{F}_1$ 和 $\boldsymbol{E}_1$ 是三对角矩阵. 假定每个分块都是 $n \times n$ 的, 也假定通过 (4.5.3) 和 (4.5.5) 求解 (4.5.9) 是 "可靠的". 注意:

$$\begin{aligned}
\boldsymbol{U}_1 &= \boldsymbol{D}_1, && \text{(对角)}\\
\boldsymbol{L}_1 &= \boldsymbol{E}_1\boldsymbol{U}_1^{-1}, && \text{(三对角)}\\
\boldsymbol{U}_2 &= \boldsymbol{D}_2 - \boldsymbol{L}_1\boldsymbol{F}_1, && \text{(五对角)}\\
\boldsymbol{y}_1 &= \boldsymbol{b}_1,\\
\boldsymbol{y}_2 &= \boldsymbol{b}_2 - \boldsymbol{E}_1(\boldsymbol{D}_1^{-1}\boldsymbol{y}_1),\\
\boldsymbol{U}_2\boldsymbol{x}_2 &= \boldsymbol{y}_2,\\
\boldsymbol{D}_1\boldsymbol{x}_1 &= \boldsymbol{y}_1 - \boldsymbol{F}_1\boldsymbol{x}_2.
\end{aligned}$$

于是, 由最初的带状分块矩阵通过简单的 $n$ 阶矩阵计算便可求得解.

另一方面, 将带状高斯消去法机械地应用于 (4.5.9) 会因方程带宽为 $n+1$ 而耗费大量不必要的计算和存储空间. 然而, 我们指出, 通过置换矩阵

$$\boldsymbol{P} = [e_1, e_{n+1}, e_2, \cdots, e_n, e_{2n}] \tag{4.5.10}$$

将方程组行和列进行置换, 可以发现 (以 $n = 5$ 为例):

$$\boldsymbol{PAP}^{\mathrm{T}} = \begin{bmatrix}
\times & \times & 0 & \times & 0 & 0 & 0 & 0 & 0 & 0\\
\times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & \times & \times & \times & 0 & \times & 0 & 0 & 0 & 0\\
\times & 0 & \times & \times & \times & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & \times & \times & \times & 0 & \times & 0 & 0\\
0 & 0 & \times & 0 & \times & \times & \times & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & \times & \times & \times & 0 & \times\\
0 & 0 & 0 & 0 & \times & 0 & \times & \times & \times & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times\\
0 & 0 & 0 & 0 & 0 & 0 & \times & 0 & \times & \times
\end{bmatrix}.$$

此矩阵的上带宽和下带宽都为 3, 因此对矩阵 $\boldsymbol{A}$ 的该置换形式用带状高斯消去法就得到一个很合理的算法.

压缩带宽的置换是很重要的问题. 请参考 George and Liu (1981, 第 4 章). 有关求解分块三对角方程组的详细讨论, 请参阅 Varah (1972) 和 George (1974).

### 4.5.4 分块循环约化法

接下来给出*分块循环约化法*, 它可用来求解分块三对角方程组 (4.5.1) 的某些很重要的特例. 为简便起见, 假定 $\boldsymbol{A}$ 具有如下形式:

$$\boldsymbol{A}=\begin{bmatrix} \boldsymbol{D} & \boldsymbol{F} & & \cdots & \boldsymbol{0} \\ \boldsymbol{F} & \boldsymbol{D} & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{F} \\ \boldsymbol{0} & \cdots & & \boldsymbol{F} & \boldsymbol{D} \end{bmatrix} \in \mathbb{R}^{nq\times nq}, \tag{4.5.11}$$

其中, $\boldsymbol{F}$ 和 $\boldsymbol{D}$ 是 $q\times q$ 矩阵且满足 $\boldsymbol{DF}=\boldsymbol{FD}$. 同时假定 $n=2^k-1$. 这些条件在许多重要的应用中都是成立的, 如矩形域上 Poisson 方程的离散化. 在这种情形下

$$\boldsymbol{D}=\begin{bmatrix} 4 & -1 & & \cdots & 0 \\ -1 & 4 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & -1 \\ 0 & \cdots & & -1 & 4 \end{bmatrix} \tag{4.5.12}$$

且 $\boldsymbol{F}=-\boldsymbol{I}_q$. 整数 $n$ 由网格的大小决定, 通常可选为具有形式 $n=2^k-1$ (当维数不是这种形式时, Sweet (1977) 给出了处理方法).

循环约化法的基本思想是将问题的维数反复地减半, 直到得到一个关于未知子向量 $\boldsymbol{x}_{2^{k-1}}$ 的 $q\times q$ 方程组. 这个方程组通过标准方法解出. 前面消去的 $x_i$ 再通过向后迭代法解出.

考虑 $n=7$ 情形足以说明方法的一般情形:

$$\begin{aligned}
\boldsymbol{b}_1 &= \boldsymbol{D}\boldsymbol{x}_1+\boldsymbol{F}\boldsymbol{x}_2, \\
\boldsymbol{b}_2 &= \boldsymbol{F}\boldsymbol{x}_1+\boldsymbol{D}\boldsymbol{x}_2+\boldsymbol{F}\boldsymbol{x}_3, \\
\boldsymbol{b}_3 &= \boldsymbol{F}\boldsymbol{x}_2+\boldsymbol{D}\boldsymbol{x}_3+\boldsymbol{F}\boldsymbol{x}_4, \\
\boldsymbol{b}_4 &= \boldsymbol{F}\boldsymbol{x}_3+\boldsymbol{D}\boldsymbol{x}_4+\boldsymbol{F}\boldsymbol{x}_5, \\
\boldsymbol{b}_5 &= \boldsymbol{F}\boldsymbol{x}_4+\boldsymbol{D}\boldsymbol{x}_5+\boldsymbol{F}\boldsymbol{x}_6, \\
\boldsymbol{b}_6 &= \boldsymbol{F}\boldsymbol{x}_5+\boldsymbol{D}\boldsymbol{x}_6+\boldsymbol{F}\boldsymbol{x}_7, \\
\boldsymbol{b}_7 &= \boldsymbol{F}\boldsymbol{x}_6+\boldsymbol{D}\boldsymbol{x}_7,
\end{aligned} \tag{4.5.13}$$

对于 $i=2,4$ 和 6, 我们分别用 $\boldsymbol{F},-\boldsymbol{D}$ 和 $\boldsymbol{F}$ 去乘第 $i-1$ 个、第 $i$ 个和第 $i+1$ 个方程, 然后将得到的方程相加:

$$\begin{aligned}
(2\boldsymbol{F}^2-\boldsymbol{D}^2)\boldsymbol{x}_2+\boldsymbol{F}^2\boldsymbol{x}_4 &= \boldsymbol{F}(\boldsymbol{b}_1+\boldsymbol{b}_3)-\boldsymbol{D}\boldsymbol{b}_2, \\
\boldsymbol{F}^2\boldsymbol{x}_2+(2\boldsymbol{F}^2-\boldsymbol{D}^2)\boldsymbol{x}_4+\boldsymbol{F}^2\boldsymbol{x}_6 &= \boldsymbol{F}(\boldsymbol{b}_3+\boldsymbol{b}_5)-\boldsymbol{D}\boldsymbol{b}_4, \\
\boldsymbol{F}^2\boldsymbol{x}_4+(2\boldsymbol{F}^2-\boldsymbol{D}^2)\boldsymbol{x}_6 &= \boldsymbol{F}(\boldsymbol{b}_5+\boldsymbol{b}_7)-\boldsymbol{D}\boldsymbol{b}_6.
\end{aligned}$$

这样, 用这种技巧我们已将下标为奇数的 $\boldsymbol{x}_i$ 消去, 得到一个约化的分块三对角方程组, 其形式为

$$\begin{aligned}
\boldsymbol{D}^{(1)}\boldsymbol{x}_2+\boldsymbol{F}^{(1)}\boldsymbol{x}_4 &= \boldsymbol{b}_2^{(1)}, \\
\boldsymbol{F}^{(1)}\boldsymbol{x}_2+\boldsymbol{D}^{(1)}\boldsymbol{x}_4+\boldsymbol{F}^{(1)}\boldsymbol{x}_6 &= \boldsymbol{b}_4^{(1)}, \\
\boldsymbol{F}^{(1)}\boldsymbol{x}_4+\boldsymbol{D}^{(1)}\boldsymbol{x}_6 &= \boldsymbol{b}_6^{(1)}.
\end{aligned}$$

其中, $\boldsymbol{D}^{(1)}=2\boldsymbol{F}^2-\boldsymbol{D}^2$ 与 $\boldsymbol{F}^{(1)}=\boldsymbol{F}^2$ 可互换. 应用如上相同的消去方法, 将这三个方程分别乘以 $\boldsymbol{F}^{(1)},-\boldsymbol{D}^{(1)}$ 和 $\boldsymbol{F}^{(1)}$. 将这些变换后的方程相加, 得到一个方程:

$$(2[\boldsymbol{F}^{(1)}]^2-\boldsymbol{D}^{(1)^2})\boldsymbol{x}_4=\boldsymbol{F}^{(1)}(\boldsymbol{b}_2^{(1)}+\boldsymbol{b}_6^{(1)})-\boldsymbol{D}^{(1)}\boldsymbol{b}_4^{(1)},$$

将其记作

$$\boldsymbol{D}^{(2)}\boldsymbol{x}_4=\boldsymbol{b}^{(2)}.$$

这样循环约化便完成了. 现在求解这个 (小的)$q\times q$ 方程组得到向量 $\boldsymbol{x}_4$. 向量 $\boldsymbol{x}_2$ 和 $\boldsymbol{x}_6$ 可通过求解方程组

$$\boldsymbol{D}^{(1)}\boldsymbol{x}_2=\boldsymbol{b}_2^{(1)}-\boldsymbol{F}^{(1)}\boldsymbol{x}_4,$$

$$\boldsymbol{D}^{(1)}\boldsymbol{x}_6=\boldsymbol{b}_6^{(1)}-\boldsymbol{F}^{(1)}\boldsymbol{x}_4$$

而解得. 最后, 应用 (4.5.13) 中第 1, 3, 5, 7 个方程来分别计算 $\boldsymbol{x}_1,\boldsymbol{x}_3,\boldsymbol{x}_5$ 和 $\boldsymbol{x}_7$.

对一般的 $n=2^k-1$ 的情况, 我们令 $\boldsymbol{D}^{(0)}=\boldsymbol{D},\boldsymbol{F}^{(0)}=\boldsymbol{F},\boldsymbol{b}^{(0)}=\boldsymbol{b}$, 然后进行如下计算:

**for** $p=1:k-1$
  $F^{(p)}=[F^{(p-1)}]^2$
  $D^{(p)}=2F^{(p)}-[D^{(p-1)}]^2$
  $r=2^p$
  **for** $j=1:2^{k-p}-1$ (4.5.14)
    $b_{jr}^{(p)}=F^{(p-1)}(b_{jr-r/2}^{(p-1)}+b_{jr+r/2}^{(p-1)})-D^{(p-1)}b_{jr}^{(p-1)}$
  **end**
**end**

然后, $\boldsymbol{x}_i$ 由下面的计算过程解出:

从 $D^{(k-1)}x_{2^{k-1}}=b_1^{(k-1)}$ 解出 $x_{2^{k-1}}$
**for** $p=k-2:-1:0$
  $r=2^p$
  **for** $j=1:2^{k-p-1}$ (4.5.15)
    **if** $j=1$
      $c=b_{(2j-1)r}^{(p)}-F^{(p)}x_{2jr}$
    **elseif** $j=2^{k-p-1}$
      $c=b_{(2j-1)r}^{(p)}-F^{(p)}x_{(2j-2)r}$
    **else**
      $c=b_{(2j-1)r}^{(p)}-F^{(p)}(x_{2jr}+x_{(2j-2)r})$
    **end**
    从 $D^{(p)}x_{(2j-1)r}=c$ 解出 $x_{(2j-1)r}$
  **end**
**end**

执行这些递推的工作量在很大程度上依赖于 $\boldsymbol{D}^{(p)}$ 和 $\boldsymbol{F}^{(p)}$ 的稀疏性. 在较坏情况下, 当这些矩阵是满的时, 总的 flop 数为 $\log(n)q^3$ 量级, 为保证约化过程的稳定性, 执行时须加小心. 更详细的讨论, 请见 Buneman(1969).

**例 4.5.1** 假设在 (4.5.14) 中, $q=1, \boldsymbol{D}=(4), \boldsymbol{F}=(-1)$, 我们需要求解

$$\begin{bmatrix} 4 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \\ 10 \\ 12 \\ 22 \end{bmatrix}.$$

通过执行 (4.5.15) 得到约化的方程组:

$$\begin{bmatrix} -14 & 1 & 0 \\ 1 & -14 & 1 \\ 0 & 1 & -14 \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \end{bmatrix} = \begin{bmatrix} -24 \\ -48 \\ -80 \end{bmatrix}, \quad p=1,$$

和

$$[-194] = [x_4][-776], \quad p=2.$$

则由 (4.5.16) 可得到 $x_i$:

$$\begin{aligned} p=2: &\quad x_4 = 4, \\ p=1: &\quad x_2 = 2, \quad x_6 = 6, \\ p=0: &\quad x_1 = 1, \quad x_3 = 3, \quad x_5 = 5, \quad x_7 = 7. \end{aligned}$$

循环约化是分而治之算法的一个例子. 1.3.8 节和 8.6 节讨论了其他分而治之方法.

### 4.5.5 Kronecker 积方程组

如果 $\boldsymbol{B} \in \mathbb{R}^{m\times n}, \boldsymbol{C} \in \mathbb{R}^{p\times q}$, 则定义 Kronecker 积为

$$\boldsymbol{A} = \boldsymbol{B} \otimes \boldsymbol{C} = \begin{bmatrix} b_{11}\boldsymbol{C} & b_{12}\boldsymbol{C} & \cdots & b_{1n}\boldsymbol{C} \\ b_{21}\boldsymbol{C} & b_{22}\boldsymbol{C} & \cdots & b_{2n}\boldsymbol{C} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1}\boldsymbol{C} & b_{m2}\boldsymbol{C} & \cdots & b_{mn}\boldsymbol{C} \end{bmatrix}.$$

因此, $\boldsymbol{A}$ 是一个 $m\times n$ 分块矩阵, 其 $(i,j)$ 块为 $b_{ij}\boldsymbol{C}$. Kronecker 积是与许多网格离散化问题相关联产生的, 且贯穿于信号处理中. Kronecker 积所满足的一些重要性质包括

$$(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D}) = \boldsymbol{AC}\otimes\boldsymbol{BD}, \tag{4.5.16}$$

$$(\boldsymbol{A}\otimes\boldsymbol{B})^{\mathrm{T}} = \boldsymbol{A}^{\mathrm{T}}\otimes\boldsymbol{B}^{\mathrm{T}}, \tag{4.5.17}$$

$$(\boldsymbol{A}\otimes\boldsymbol{B})^{-1}=\boldsymbol{A}^{-1}\otimes\boldsymbol{B}^{-1}, \tag{4.5.18}$$

其中假定所有的乘法运算都有意义.

与 Kronecker 积相关的是 “vec” 运算

$$\boldsymbol{X}\in\mathbb{R}^{m\times n}\Leftrightarrow \mathrm{vec}(\boldsymbol{X})=\begin{bmatrix} X(:,1)\\ \vdots \\ X(:,n)\end{bmatrix}\in\mathbb{R}^{mn}.$$

于是, 矩阵的 vec 运算就是它的列向量 “堆” 起来. 可以证明

$$\boldsymbol{Y}=\boldsymbol{C}\boldsymbol{X}\boldsymbol{B}^{\mathrm{T}}\Leftrightarrow \mathrm{vec}(\boldsymbol{Y})=(\boldsymbol{B}\otimes\boldsymbol{C})\mathrm{vec}(\boldsymbol{X}). \tag{4.5.19}$$

因此, 求解 Kronecker 积方程组

$$(\boldsymbol{B}\otimes\boldsymbol{C})\boldsymbol{x}=\boldsymbol{d}$$

等价于求解关于 $\boldsymbol{X}$ 的矩阵方程 $\boldsymbol{C}\boldsymbol{X}\boldsymbol{B}^{\mathrm{T}}=\boldsymbol{D}$, 其中, $\boldsymbol{x}=\mathrm{vec}(\boldsymbol{X}),\boldsymbol{d}=\mathrm{vec}(\boldsymbol{D})$. 这样做同时获得了效率. 为说明这点, 假定 $\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{n\times n}$ 是对称正定的. 如果把 $\boldsymbol{A}=\boldsymbol{B}\otimes\boldsymbol{C}$ 看作一般矩阵, 进行分解以求 $x$, 则由于 $\boldsymbol{B}\otimes\boldsymbol{C}\in\mathbb{R}^{n^2\times n^2}$, 所需工作量为 $O(n^6)$ 个 flop. 另一方面, 如按下面方法求解:

(1) 计算 Cholesky 分解 $\boldsymbol{B}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 和 $\boldsymbol{C}=\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}$;

(2) 利用 $\boldsymbol{G}$ 从 $\boldsymbol{B}\boldsymbol{Z}=\boldsymbol{D}^{\mathrm{T}}$ 解出 $\boldsymbol{Z}$;

(3) 利用 $\boldsymbol{H}$ 从 $\boldsymbol{C}\boldsymbol{X}=\boldsymbol{Z}^{\mathrm{T}}$ 解出 $\boldsymbol{X}$;

(4) $x=\mathrm{vec}(\boldsymbol{X})$.

则需 $O(n^3)$ 个 flop. 注意到

$$\boldsymbol{B}\otimes\boldsymbol{C}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\otimes\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}=(\boldsymbol{G}\otimes\boldsymbol{H})(\boldsymbol{G}\otimes\boldsymbol{H})^{\mathrm{T}}$$

是 $\boldsymbol{B}\otimes\boldsymbol{C}$ 的 Cholesky 分解, 这是因为两个下三角形矩阵的 Kronecker 积仍是下三角形矩阵. 因此上述四步求解方法是应用于 $\boldsymbol{B}\otimes\boldsymbol{C}$ 的利用结构的 Cholesky 法.

有必要指出, 如果 $\boldsymbol{B}$ 是稀疏的, 则在块层次上 $\boldsymbol{B}\otimes\boldsymbol{C}$ 也有相同的稀疏结构. 例如, 如果 $\boldsymbol{B}$ 是三对角的, 则 $\boldsymbol{B}\otimes\boldsymbol{C}$ 是分块三对角的.

## 习　题

**4.5.1** 证明分块对角占优矩阵是非奇异的.

**4.5.2** 验证由 (4.5.6) 可推出 (4.5.7) 和 (4.5.8).

**4.5.3** 设将分块循环约化法应用于 (4.5.12) 所给定的 $\boldsymbol{D}$, 且有 $\boldsymbol{F}=-\boldsymbol{I}q$, 则产生的矩阵 $\boldsymbol{F}^{(p)}$ 和 $\boldsymbol{D}^{(p)}$ 具有什么样的带状结构?

**4.5.4** 假设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是非奇异的, 且线性方程组 $\boldsymbol{A}\boldsymbol{z}=\boldsymbol{b}$ 和 $\boldsymbol{A}\boldsymbol{y}=\boldsymbol{g}$ 有解, 其中 $\boldsymbol{b},\boldsymbol{g}=\mathbb{R}^n$ 给定, 试在 $O(n)$ 个 flop 内解方程组

$$\begin{bmatrix}\boldsymbol{A} & \boldsymbol{g}\\ \boldsymbol{h}^{\mathrm{T}} & \alpha\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \mu\end{bmatrix}=\begin{bmatrix}\boldsymbol{b}\\ \beta\end{bmatrix},$$

其中 $\alpha, \beta \in \mathbb{R}$ 和 $\boldsymbol{h} \in \mathbb{R}^n$ 给定, 系数矩阵 $\boldsymbol{A}_+$ 非奇异. 寻找一个快速解法是否成功是一个复杂的问题, 它取决于 $\boldsymbol{A}$ 和 $\boldsymbol{A}_+$ 的条件数以及其他因素.

**4.5.5**　证明 (4.5.16)~(4.5.19).

**4.5.6**　说明如何由 $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 的 SVD 分解构造 $\boldsymbol{B} \otimes \boldsymbol{C}$ 的 SVD 分解.

**4.5.7**　如果 $\boldsymbol{A}, \boldsymbol{B}$ 和 $\boldsymbol{C}$ 均为矩阵, 则可以证明 $(\boldsymbol{A} \otimes \boldsymbol{B}) \otimes \boldsymbol{C} = \boldsymbol{A} \otimes (\boldsymbol{B} \otimes \boldsymbol{C})$, 于是我们可以只用 $\boldsymbol{A} \otimes \boldsymbol{B} \otimes \boldsymbol{C}$ 代表这个矩阵. 说明当 $\boldsymbol{A}, \boldsymbol{B}$ 和 $\boldsymbol{C}$ 均对称正定矩阵时, 如何求解 $(\boldsymbol{A} \otimes \boldsymbol{B} \otimes \boldsymbol{C})\boldsymbol{x} = \boldsymbol{d}$.

## 本节注释与参考文献

下述文章对分块矩阵运算的许多细节做了详尽探讨:

J. M. Varah (1972). "On the Solution of Block-Tridiagonal Systems Arising from Certain Finite-Difference Equations," *Math. Comp.* 26, 859–868.

J. A. George (1974). "On Block Elimination for Sparse Linear Systems," *SIAM J. Num. Anal.* 11, 585–603.

R. Fourer (1984). "Staircase Matrices and Systems," *SIAM Review* 26, 1–71.

M. L. Merriam (1985). "On the Factorization of Block Tridiagonals With Storage Constraints," *SIAM J. Sci. and Stat. Comp.* 6, 182–192.

分块对角占优矩阵的性质和其许多推论是下述文章的主题:

D. G. Feingold and R. S. Varga (1962). "Block Diagonally Dominant Matrices and Generalizations of the Gershgorin Circle Theorem," *Pacific J. Math.* 12, 1241–1250.

涉及循环约化思想的早期方法见于:

R. W. Hockney (1965). "A Fast Direct Solution of Poisson's Equation Using Fourier Analysis," *J. ACM* 12, 95–113.

B. L. Buzbee, G. H. Golub, and C. W. Nielson (1970). "On Direct Methods for Solving Poisson's Equations," *SIAM J. Num. Anal.* 7, 627–656.

右端向量的累积必须仔细处理, 否则会产生精确严重失去. 解决这个问题的一个稳定方法见:

O. Buneman (1969). "A Compact Non-Iterative Poisson Solver," Report 294, Stanford University Institute for Plasma Research, Stamford, California.

其他关于循环约化的文献包括:

F. W. Dorr (1970). "The Direct Solution of the Discrete Poisson Equation on a Rectangle," *SIAM Review* 12, 248–263.

B. L. Buzbee, F. W. Dorr, J. A. George, and G. H. Golub (1971). "The Direct Solution of the Discrete Poisson Equation on Irregular Regions," *SIAM J. Num. Anal.* 8, 722–736.

F. W. Dorr (1973). "The Direct Solution of the Discrete Poisson Equation in $O(n^2)$ Operations," *SIAM Review* 15, 412–415.

P. Concus and G. H. Golub (1973). "Use of Fast Direct Methods for the Efficient Numerical

Solution of Nonseparable Elliptic Equations," *SIAM J. Num. Anal.* 10, 1103–1120.

B. L. Buzbee and F. W. Dorr (1974). "The Direct Solution of the Biharmonic Equation on Rectangular Regions and the Poisson Equation on Irregular Regions," *SIAM J. Num. Anal.* 11, 753–763.

D. Heller (1976). "Some Aspects of the Cyclic Reduction Algorithm for Block Tridiagonal Linear Systems," *SIAM J. Num. Anal.* 13, 484–496.

循环约化的一些总结和推广见下面的文献:

P. N. Swarztrauber and R. A. Sweet (1973). "The Direct Solution of the Discrete Poisson Equation on a Disk," *SIAM J. Num. Anal.* 10, 900–907.

R. A. Sweet (1974). "A Generalized Cyclic Reduction Algorithm," *SIAM J. Num. Anal.* 11, 506–520.

M. A. Diamond and D. L. V. Ferreira (1976). "On a Cyclic Reduction Method for the Solution of Poisson's Equation," *SIAM J. Num. Anal.* 13, 54–70.

R. A. Sweet (1977). "A Cyclic Reduction Algorithm for Solving Block Tridiagonal Systems of Arbitrary Dimension," *SIAM J. Num. Anal.* 14, 706–720.

P. N. Swarztrauber and R. Sweet (1989). "Vector and Parallel Methods for the Direct Solution of Poisson's Equation," *J. Comp. Appl. Math.* 27, 241–263.

S. Bondeli and W. Gander (1994). "Cyclic Reduction for Special Tridiagonal Systems," *SIAM J. Matrix Anal. Appl.* 15, 321–330.

对于一些由椭圆形偏微分方程导出的矩阵来说, 分块消去法对应于其网格的自然的运算, 此方法的一个典型的例子是嵌套分割方法, 见:

A. George (1973). "Nested Dissection of a Regular Finite Element Mesh," *SIAM J. Num. Anal.* 10, 345–363.

我们指出如下的综述性文章:

J. R. Bunch (1976). "Block Methods for Solving Sparse Linear Systems," in *Sparse Matrix Computations*, J. R. Bunch and D. J. Rose (eds), Academic Press, New York.

习题 4.5.4 中给出的加边线性方程组在下述文章中讨论:

W. Govaerts and J. D. Pryce (1990). "Block Elimination with One Iterative Refinement Solves Bordered Linear Systems Accurately," *BIT* 30, 490–507.

W. Govaerts (1991). "Stable Solvers and Block Elimination for Bordered Systems," *SIAM J. Matrix Anal. Appl.* 12, 469–483.

W. Govaerts and J. D. Pryce (1993). "Mixed Block Elimination for Linear Systems with Wider Borders," *IMA J. Num. Anal,* 13, 161–180.

Kronecker 积的参考文献包含:

H. C. Andrews and J. Kane (1970). "Kronecker Matrices, Computer Implementation, and Generalized Spectra," *J. Assoc. Comput, Mach.* 17, 260–268.

C. de Boor (1979). "Efficient Computer Manipulation of Tensor Products," *ACM Trans. Math. Soft.* 5, 173–182.

A. Graham (1981). *Kronecker Products and Matrix Calculus with Applications*, Ellis Horwood Ltd., Chichester, England.

H. V. Henderson and S. R. Searle (1981). "The Vec-Permutation Matrix, The Vec Operator, and Kronecker Products: A Review," *Linear and Multilinear Algebra*, 9, 271–288.

P. A. Regalia and S. Mitra (1989). "Kronecker Products, Unitary Matrices, and Signal Processing Applications," *SIAM Review* 31, 586–613.

## 4.6 Vandermonde 方程组和 FFT

假定 $x(0:n) \in \mathbb{R}^{n+1}$, 形如

$$\boldsymbol{V} = V(x_0.\cdots,x_n) = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_0 & x_1 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_0^n & x_1^n & \cdots & x_n^n \end{bmatrix}$$

的矩阵 $\boldsymbol{V} \in \mathbb{R}^{(n+1)\times(n+1)}$ 被称作 Vandermonde 矩阵. 本节我们讨论如何在 $O(n^2)$ 个 flop 内求解方程组 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{a} = \boldsymbol{f} = \boldsymbol{f}(0:n)$ 和 $\boldsymbol{V}\boldsymbol{z} = \boldsymbol{b} = \boldsymbol{b}(0:n)$. 同时简单介绍了离散的 Fourier 变换. 这种特殊且很重要的 Vandermonde 方程组具有递归的块结构并且可在 $O(n\log n)$ 个 flop 内解出. 本节中的向量和矩阵的下标均从 0 开始标记.

### 4.6.1 多项式插值: $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{a} = \boldsymbol{f}$

Vandermonde 方程组出现在许多近似和插值问题中. 实际上, 快速求得 Vandermonde 方程组的关键是要认识到解 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{a} = \boldsymbol{f}$ 等价于多项式插值. 这是因为, 如果 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{a} = \boldsymbol{f}$ 且有

$$p(x) = \sum_{j=0}^{n} a_j x^j, \tag{4.6.1}$$

则对于 $i = 0:n$, 有 $p(x_i) = f_i$.

如果 $x_i$ 是互异的, 则可对插值点 $(x_0, f_0), \cdots, (x_n, f_n)$ 构造一个唯一的 $n$ 阶多项式. 因此只要 $x_i$ 是互异的, 则 $\boldsymbol{V}$ 是非奇异的. 在整个这一节我们都假定此条件成立.

计算 (4.6.1) 中 $a_j$ 的第一步是计算插值多项式 $p$ 的牛顿表达式:

$$p(x) = \sum_{k=0}^{n} c_k \left( \prod_{i=0}^{k-1} (x - x_i) \right). \tag{4.6.2}$$

常数 $c_k$ 是均差, 可按下面步骤来确定:

$c(0:n) = f(0:n)$

**for** $k = 0:n-1$

  **for** $i = n:-1:k+1$  (4.6.3)

$c_i = (c_i - c_{i-1})/(x_i - x_{i-k-1})$
**end**
**end**

参见 Conte and de Boor (1980, 第 2 章).

下一步任务是由 $c(0:n)$ 来产生 $a(0:n)$, 通过迭代

$p_n(x) = c_n$
**for** $k = n-1:-1:0$
$p_k(x) = c_k + (x - x_k)p_{k+1}(x)$
**end**

来定义多项式 $p_n(x), \cdots, p_0(x)$, 注意到 $p_0(x) = p(x)$. 写出

$$p_k(x) = a_k^{(k)} + a_{k+1}^{(k)}x + \cdots + a_n^{(k)}x^{n-k},$$

令方程 $p_k = c_k + (x - x_k)p_{k+1}$ 中 $x$ 的同次幂项相等, 得到下面求系数 $a_i^{(k)}$ 的递推关系式:

$a_n^{(n)} = c_n$
**for** $k = n-1:-1:0$
$a_k^{(k)} = c_k - x_k a_{k+1}^{(k+1)}$
**for** $i = k+1:n-1$
$a_i^{(k)} = a_i^{(k+1)} - x_k a_{i+1}^{(k+1)}$
**end**
$a_n^{(k)} = a_n^{(k+1)}$
**end**

于是, 系数 $a_i = a_i^{(0)}$ 可按下面步骤求得

$a(0:n) = c(0:n)$
**for** $k = n-1:-1:0$
**for** $i = k:n-1$
$a_i = a_i - x_k a_{i+1}$ (4.6.4)
**end**
**end**

将此迭代法与 (4.6.3) 合并得到下面的算法.

**算法 4.6.1** 给定 $x(0:n) \in \mathbb{R}^{n+1}$, 其元素互异, $f = f(0:n) \in \mathbb{R}^{n+1}$, 本算法求出 Vandermonde 方程组 $V(x_0, \cdots, x_n)^{\mathrm{T}}a = f$ 的解 $a = a(0:n)$, 并将其存储在 $f$ 中.

**for** $k = 0:n-1$
**for** $i = n:-1:k+1$
$f(i) = (f(i) - f(i-1))/(x(i) - x(i-k-1))$

```
        end
    end
    for k = n-1 : -1 : 0
        for i = k : n-1
            f(i) = f(i) - f(i+1)x(k)
        end
    end
```

本算法中需 $5n^2/2$ 个 flop.

**例 4.6.1** 假设用算法 4.6.1 来求解

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 26 \\ 58 \\ 112 \end{bmatrix},$$

第一次 $k$ 循环计算得到 $p(x)$ 的牛顿形式

$$p(x) = 10 + 16(x-1) + 8(x-1)(x-2) + (x-1)(x-2)(x-3),$$

第二次 $k$ 循环由 $[10\ \ 16\ \ 8\ \ 1]^{\mathrm{T}}$ 解得 $a = [4\ \ 3\ \ 2\ \ 1]^{\mathrm{T}}$.

### 4.6.2 形如 $Vz = b$ 的方程组

现在考虑方程组 $Vz = b$. 为导出解此问题的一个高效算法, 我们用矩阵–向量语言来刻画算法 4.6.1. 定义下双对角矩阵 $L_k(\alpha) \in \mathbb{R}^{(n+1)\times(n+1)}$ 如下:

$$L_k(\alpha) = \left[\begin{array}{c|ccccc} I_k & & & 0 & & \\ \hline & 1 & & \cdots & & 0 \\ & -\alpha & 1 & & & \\ & & \ddots & \ddots & & \\ 0 & \vdots & & \ddots & \ddots & \vdots \\ & & & & \ddots\ \ 1 & \\ & 0 & & \cdots & -\alpha & 1 \end{array}\right].$$

并且定义对角矩阵 $D_k$ 为

$$D_k = \mathrm{diag}(\underbrace{1, \cdots, 1}_{k+1}, x_{k+1} - x_0, \cdots, x_n - x_{n-k-1}).$$

由这些定义和 (4.6.3) 容易推知, 如 $f = f(0:n)$ 和 $c = c(0:n)$ 是均差向量, 则 $c = U^{\mathrm{T}} f$, 其中 $U$ 是上三角形矩阵, 其定义为

$$U^{\mathrm{T}} = D_{n-1}^{-1} L_{n-1}(1) \cdots D_0^{-1} L_0(1).$$

类似地, 由 (4.6.4) 有

$$a = L^{\mathrm{T}} c,$$

其中 $L$ 是单位下三角形矩阵, 其定义为

$$L^{\mathrm{T}} = L_0(x_0)^{\mathrm{T}} \cdots L_{n-1}(x_{n-1})^{\mathrm{T}}.$$

于是, $a = L^{\mathrm{T}}U^{\mathrm{T}}f$, 其中 $V^{-\mathrm{T}} = L^{\mathrm{T}}U^{\mathrm{T}}$. 换句话说, 算法 4.6.1 无形中利用了 $V^{-1}$ 的 UL 分解来求解 $V^{\mathrm{T}}a = f$.

因此, 方程组 $Vz = b$ 的解为

$$\begin{aligned} z &= V^{-1}b = U(Lb) \\ &= (L_0(1)^{\mathrm{T}}D_0^{-1} \cdots L_{n-1}(1)^{\mathrm{T}}D_{n-1}^{-1})(L_{n-1}(x_{n-1}) \cdots L_0(x_0)b). \end{aligned}$$

通过以上观察得出下面的算法.

**算法 4.6.2** 给定 $x(0:n) \in \mathbb{R}^{n+1}$, 其元素互异且 $b = b(0:n) \in \mathbb{R}^{n+1}$, 本算法以 Vandermonde 方程组 $V(x_0, \cdots, x_n)z = b$ 的解 $z = z(0:n)$ 来覆盖 $b$.

```
for k = 0 : n − 1
    for i = n : −1 : k + 1
        b(i) = b(i) − x(k)b(i − 1)
    end
end
for k = n − 1 : −1 : 0
    for i = k + 1 : n
        b(i) = b(i)/(x(i) − x(i − k − 1))
    end
    for i = k : n − 1
        b(i) = b(i) − b(i + 1)
    end
end
```

本算法需 $5n^2/2$ 个 flop.

**例 4.6.2** 假设用算法 4.6.2 来解

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 3 \\ 35 \end{bmatrix},$$

第一次 $k$ 循环计算向量

$$L_3(3)L_2(2)L_1(1) \begin{bmatrix} 0 \\ -1 \\ 3 \\ 35 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 6 \\ 6 \end{bmatrix},$$

第二次 $k$ 循环则计算

$$L_0(1)^{\mathrm{T}}D_0^{-1}L_1(1)^{\mathrm{T}}D_1^{-1}L_2(1)^{\mathrm{T}}D_2^{-1}\begin{bmatrix}0\\-1\\3\\35\end{bmatrix}=\begin{bmatrix}3\\-4\\0\\1\end{bmatrix}.$$

### 4.6.3　稳定性

Björck and Pereyra (1970) 对算法 4.6.1 和算法 4.6.2 作了讨论和分析. 他们的结论是这两个算法经常得到令人吃惊的准确解, 即使 $\boldsymbol{V}$ 病态也是如此. 他们指出当插值点集上增加一个新插值对 $(x_{n+1}, f_{n+1})$ 时, 如何修正已有解. 同时还给出了如何求解汇合型 Vandermonde 方程组, 它涉及形如

$$\boldsymbol{V}=V(x_0,x_1,x_2,x_3)=\begin{bmatrix}1&1&0&1\\x_0&x_1&1&x_3\\x_0^2&x_1^2&2x_2^2&x_3^2\\x_0^3&x_1^3&3x_2^2&x_3^3\end{bmatrix}$$

的矩阵.

### 4.6.4　快速 Fourier 变换

$n$ 阶的离散 Fourier 变换阵 (DFT) 定义为

$$\boldsymbol{F}_n=(f_{ik}),\quad f_{jk}=\omega_n^{jk},$$

其中

$$\omega_n=\exp(-2\pi \mathrm{i}/n)=\cos(2\pi/n)-\mathrm{i}\cdot\sin(2\pi/n).$$

由于 $\omega_n^n=1$, 故参数 $\omega_n$ 是 1 的 $n$ 次方根. 当 $n=4$ 时, $\omega_4=-\mathrm{i}$, 且有

$$\boldsymbol{F}_4=\begin{bmatrix}1&1&1&1\\1&\omega_4&\omega_4^2&\omega_4^3\\1&\omega_4^2&\omega_4^4&\omega_4^6\\1&\omega_4^3&\omega_4^6&\omega_4^9\end{bmatrix}=\begin{bmatrix}1&1&1&1\\1&-\mathrm{i}&-1&\mathrm{i}\\1&-1&1&-1\\1&\mathrm{i}&-1&-\mathrm{i}\end{bmatrix}.$$

如果 $\boldsymbol{x}\in\mathbb{C}^n$, 则其 DFT 为向量 $\boldsymbol{F}_n\boldsymbol{x}$. DFT 在应用数学和工程中扮演着极其重要的角色.

如果 $n$ 有多个因子, 那么执行 DFT 运算可能要比常规的矩阵与向量相乘所需的 $O(n^2)$ 个 flop 要少得多. 为说明这一点我们令 $n=2^t$, 然后给出 2 基数快速 Fourier 变换 (FFT). 首先看一下偶数阶 DFT 矩阵, 对列做置换, 使下标为偶数的列都置换到矩阵的前部. 当 $n=8$ 时, 注意到 $\omega_8^{kj}=\omega_n^{kj \bmod 8}$, 我们有如下形式:

$$\boldsymbol{F}_8=\begin{bmatrix}1&1&1&1&1&1&1&1\\1&\omega&\omega^2&\omega^3&\omega^4&\omega^5&\omega^6&\omega^7\\1&\omega^2&\omega^4&\omega^6&1&\omega^2&\omega^4&\omega^6\\1&\omega^3&\omega^6&\omega&\omega^4&\omega^7&\omega^2&\omega^5\\1&\omega^4&1&\omega^4&1&\omega^4&1&\omega^4\\1&\omega^5&\omega^2&\omega^7&\omega^4&\omega&\omega^6&\omega^3\\1&\omega^6&\omega^4&\omega^2&1&\omega^6&\omega^4&\omega^2\\1&\omega^7&\omega^6&\omega^5&\omega^4&\omega^3&\omega^2&\omega\end{bmatrix},\quad \omega=\omega_8.$$

如果定义下标向量 $c = [0\ \ 2\ \ 4\ \ 6\ \ 1\ \ 3\ \ 5\ \ 7]$, 则

$$F_8(:,c) = \left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega & \omega^3 & \omega^5 & \omega^7 \\ 1 & \omega^4 & 1 & \omega^4 & \omega^2 & \omega^6 & \omega^2 & \omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 & \omega^3 & \omega & \omega^7 & \omega^5 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & \omega^2 & \omega^4 & \omega^6 & -\omega & -\omega^3 & -\omega^5 & -\omega^7 \\ 1 & \omega^4 & 1 & \omega^4 & -\omega^2 & -\omega^6 & -\omega^2 & -\omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 & -\omega^3 & -\omega & -\omega^7 & -\omega^5 \end{array}\right].$$

矩阵中的线有助于将 $F_n(0:c)$ 想象成一个 $2\times 2$ 的分块矩阵, 每块大小为 $4\times 4$. 注意到 $\omega^2 = \omega_8^2 = \omega_4$, 故

$$F_8(:,c) = \left[\begin{array}{c|c} \boldsymbol{F}_4 & \boldsymbol{\Omega}_4\boldsymbol{F}_4 \\ \hline \boldsymbol{F}_4 & -\boldsymbol{\Omega}_4\boldsymbol{F}_4 \end{array}\right],$$

其中

$$\boldsymbol{\Omega}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \omega_8 & 0 & 0 \\ 0 & 0 & \omega_8^2 & 0 \\ 0 & 0 & 0 & \omega_8^3 \end{bmatrix}.$$

于是, 如果 $\boldsymbol{x}$ 是一个 8 维向量, 则

$$\begin{aligned} \boldsymbol{F}_8\boldsymbol{x} = F(:,c)x(c) &= \left[\begin{array}{c|c} F_4 & \Omega_4F_4 \\ \hline F_4 & -\Omega_4F_4 \end{array}\right] \left[\begin{array}{c} x(0:2:7) \\ \hline x(1:2:7) \end{array}\right] \\ &= \left[\begin{array}{c|c} I & \Omega_4 \\ \hline I & -\Omega_4 \end{array}\right] \left[\begin{array}{c} F_4x(0:2:7) \\ \hline F_4x(1:2:7) \end{array}\right]. \end{aligned}$$

这样, 通过简单数乘可由 4 点的 DFT 运算 $\boldsymbol{y}_T = \boldsymbol{F}_4x(0:2:7)$ 和 $\boldsymbol{y}_B = \boldsymbol{F}_4x(1:2:7)$ 得到 8 点的 DFT 运算 $\boldsymbol{y} = \boldsymbol{F}_8\boldsymbol{x}$:

$$y(0:3) = \boldsymbol{y}_T + \boldsymbol{d}.*\boldsymbol{y}_B, \quad y(4:7) = \boldsymbol{y}_T - \boldsymbol{d}.*\boldsymbol{y}_B.$$

这里

$$d = \begin{bmatrix} 1 \\ \omega \\ \omega^2 \\ \omega^3 \end{bmatrix},$$

且 ".*" 表示向量相乘. 一般地, 如 $n = 2m$, 则 $\boldsymbol{y} = \boldsymbol{F}_n\boldsymbol{x}$ 由

$$y(0:m-1) = \boldsymbol{y}_T + \boldsymbol{d}.*\boldsymbol{y}_B, \quad y(m:n-1) = \boldsymbol{y}_T - \boldsymbol{d}.*\boldsymbol{y}_B,$$

给出, 其中

$$\boldsymbol{d} = [1,\omega,\cdots,\omega^{m-1}]^{\mathrm{T}}, \quad \boldsymbol{y}_T = \boldsymbol{F}_m x(0:2:n-1), \quad \boldsymbol{y}_B = \boldsymbol{F}_m x(1:2:n-1).$$

对于 $n=2^t$, 可以循环执行上述步骤直到 $n=1$, 此时 $\boldsymbol{F}_1\boldsymbol{x}=\boldsymbol{x}$:

**function** $y = \mathrm{FFT}(x, n)$

  **if** $n=1$

    $y = x$

  **else**

    $m = n/2; \omega = \mathrm{e}^{-2\pi\mathrm{i}/n}$

    $y_T = \mathrm{FFT}(x(0:2:n), m); y_B = \mathrm{FFT}(x(1:2:n), m)$

    $d = [1, \omega, \cdots, \omega^{m-1}]^{\mathrm{T}}; z = d.*y_B$

    $y = \begin{bmatrix} y_{\mathrm{T}} + z \\ y_{\mathrm{T}} - z \end{bmatrix}$

  **end**

这是众多 Fourier 变换算法的一种. 它有一种非循环的实现方式, 可用 $\boldsymbol{F}_n$ 的分解来描述. 实际上, 可证明 $\boldsymbol{F}_n = \boldsymbol{A}_t \cdots \boldsymbol{A}_1 \boldsymbol{P}_n$, 其中

$$\boldsymbol{A}_q = \boldsymbol{I}_r \otimes \boldsymbol{B}_L, \quad \boldsymbol{L} = 2^q, \quad r = n/\boldsymbol{L},$$

且

$$\boldsymbol{B}_L = \begin{bmatrix} I_{L/2} & \Omega_{L/2} \\ I_{L/2} & -\Omega_{L/2} \end{bmatrix}, \quad \Omega_{L/2} = \mathrm{diag}(1, \omega_L, \cdots, \omega_L^{L/2-1}).$$

矩阵 $\boldsymbol{P}_n$ 称为**反位置换** (bit reversal permutation), 我们不再详述它. (请回想 4.5.5 节中关于 Kronecker 积 "$\otimes$" 的定义). 在这样的分解下, $\boldsymbol{y} = \boldsymbol{F}_n\boldsymbol{x}$ 可按下法计算出:

$$\begin{aligned}
& x = P_n x \\
& \textbf{for } q = 1:t \\
& \quad L = 2^q, r = n/L \\
& \quad x = (I_r \otimes B_L)x \\
& \textbf{end}
\end{aligned} \tag{4.6.5}$$

矩阵 $\boldsymbol{A}_q = (\boldsymbol{I}_r \otimes \boldsymbol{B}_L)$ 的每行只有两个非零元素, 这使得 DFT 可能在 $O(n\log n)$ 个 flop 内完成. 实际上仔细的实现仅需 $5n\log_2 n$ 个 flop.

DFT 矩阵有如下性质:

$$\boldsymbol{F}_n^{-1} = \frac{1}{n}\boldsymbol{F}_n^{\mathrm{H}} = \frac{1}{n}\bar{\boldsymbol{F}}_n \tag{4.6.6}$$

即 $\boldsymbol{F}_n$ 的逆是通过对其元素取共轭, 然后除以 $n$ 得到的. 快速的 DFT 求逆, 只要将 (向前) 的 FFT 中出现的所有根 1 用其共轭复数代替, 然后除以 $n$ 即可得到.

DFT 的价值在于通过将问题变换到 Fourier 空间上 (通过 $F_n$), 使许多难解的问题简化. 然后再将 Fourier 空间上的解变换到原坐标系中 (通过 $\boldsymbol{F}_n^{-1}$) 即可得到解.

## 习 题

**4.6.1** 证明如果 $\boldsymbol{V} = \boldsymbol{V}(x_0, \cdots, x_n)$, 则

$$\det(V) = \prod_{n\geqslant i>j\geqslant 0} (x_i - x_j).$$

**4.6.2** (Cautschi 1975a) 在 $n=1$ 时, 证明下面不等式:

$$\|V^{-1}\|_\infty \leqslant \max_{0\leqslant k\leqslant n} \prod_{\substack{i=0\\ i\neq k}}^{n} \frac{1+|x_i|}{|x_k - x_i|}.$$

当复平面上所有的 $x_i$ 共线时等式成立.

**4.6.3** 假设 $w = [1, w_n, w_n^2, \cdots, w_n^{n/2-1}]$, 其中 $n = 2^t$, 用冒号表示法, 将 $[1, w_r, w_r^2, \cdots, w_r^{r/2-1}]$ 表示成 $w$ 的子向量, 其中 $r = 2^q, q = 1:t$.

**4.6.4** 证明 (4.6.6).

**4.6.5** 将 (4.6.5) 中的 $x = (I \otimes B_L)x$ 展开成两重循环, 并算出所需的 flop 数, (忽略 $x = P_n x$ 之细节).

**4.6.6** 假设 $n = 3m$, 将

$$G = [F_n(:, 0:3:n-1) \quad F_n(1:3:n-1) \quad F_n(:, 2:3:n-1)]$$

作为 $3\times 3$ 分块矩阵来考查, 找到各块 $F_m$ 的加权关系. 根据你找到的结果, 写一个类似于文中 2 基数 FFT 变换的 3 基数 FFT 变换.

## 本节注释与参考文献

我们关于 Vandermonde 线性方程组的讨论取材于以下文章:


A. Björck and V. Pereyra (1970). "Solution of Vandermonde Systems of Equations", *Math. Comp.* 24, 893–903.

A. Björck and T. Elfving (1973). "Algorithms for Confluent Vandermonds Systems", *Numer. Math.* 21, 130–137.


我们讨论的均差运算在下书的第 2 章有详细论述:


S. D. Conte and C. de Boor (1980). *Elementary Numerical Analysis: An Algorithmic Approach*, 3rd ed., McGraw-Hill, New York.


该专著中含有 Algol 程序. Vandermonde 方程组的解法的误差分析可见:


N. J. Higham (1987b). "Error Analysis of the Björck-Pereyra Algorithms for Solving Vandermonde Systems", *Numer. Math.* 50, 613–632.

N. J. Higham (1988a). "Fast Solution of Vandermonde-like Systems Involving Orthogonal Polynomials", *IMA J. Num. Anal.* 8. 473–486.

N. J. Higham (1990). "Stability Analysis of Algorithms for Solving Confluent Vandermondelike Systems", *SIAM J. Matrix Anal. Appl.* 11, 23–41.

S. G. Bartels and D. J. Higham (1992). "The Structured Sensitivity of Vandermonde-Like Systems", *Numer. Math.* 62, 17–34.

J. M. Varah (1993). "Errors and Perturbations in Vandermonde Systems", *IMA J. Num. Anal.* 13, 1–12.


关于 Vandermonde 方程组的条件数的有趣理论结果, 请见:

W. Gautschi (1975a). "Norm Estimates for Inverses of Vandermonde Matrices", *Numer. Math.* 23, 337–347.

W. Gautschi (1975b). "Optimally Conditioned Vandermonde Matrices", *Numer. Math.* 24, 1–12.

本节给出的基本算法可以推广到汇合型 Vandermonde 方程组、分块 Vandermonde 方程组和以其他基于多项式基的 Vandermonde 方程组.

G. Galimberti and V. Pereyra (1970). "Numerical Differentiation and the Solution of Multidimensional Vandermonde Systems", *Math. Comp.* 24, 357–364.

G. Galimberti and V. Pereyra (1971). "Solving Confluent Vandermonde Systems of Hermitian Type", *Numer. Math.* 18, 44–60.

H. Van de Vel (1977). "Numerical Treatment of a Generalized Vandermonde systems of Equations", *Lin. Alg. and Its Applic.* 17, 149–174.

G. H. Golub and W. P. Tang (1981). "The Block Decomposition of a Vandermonde Matrix and Its Applications", *BIT* 21, 505–617.

D. Galvetti and L. Reichel (1992). "A Chebychev-Vandermonde Solver", *Lin. Alg. and Its Applic.* 172, 219–229.

D. Calvetti and L. Reichel (1993). "Fast Inversion of Vandermonde-Like Matrices Involving Orthogonal Polynomials", *BIT* 33, 473–484.

H. Lu (1994). "Fast Solution of Confluent Vandermonde Linear Systems", *SIAM J. Matrix Anal. Appl.* 15, 1277–1289.

H. Lu (1996). "Solution of Vandermonde-like Systems and Confluent Vandermonde-like Systems", *SIAM J. Matrix Anal. Appl.* 17, 127–138.

关于 FFT 的文献, 分布广而且分散. 综述利用 Kronecker 积的领域 FFT 可参见:

C. F. Van Loan (1992), *Computational Frameworks for the Fast Fourier Transform*, SIAM Publications, Philadelphia, PA.

本书的观点是不同的 FFT 对应 DFT 矩阵不同的分解, 这些分解因子每行含有极少数的非零元, 故又称稀疏分解.

## 4.7 Toeplitz 及相关方程组

每条对角线上的元素都相同的矩阵在许多应用中出现, 这样的矩阵称为 Toeplitz 矩阵. 正式地说, 如 $\boldsymbol{T} \in \mathbb{R}^{n\times n}$, 存在标量 $r_{-n+1}, \cdots, r_0, \cdots, r_{n-1}$, 对所有 $i$ 和 $j$ 满足 $a_{ij} = r_{j-i}$, 则称 $\boldsymbol{T}$ 为 Toeplitz 矩阵. 于是,

$$\boldsymbol{T} = \begin{bmatrix} r_0 & r_1 & r_2 & r_3 \\ r_{-1} & r_0 & r_1 & r_2 \\ r_{-2} & r_{-1} & r_0 & r_1 \\ r_{-3} & r_{-2} & r_{-1} & r_0 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 7 & 6 \\ 4 & 3 & 1 & 7 \\ 0 & 4 & 3 & 1 \\ 9 & 0 & 4 & 3 \end{bmatrix}$$

是一个 Toeplitz 矩阵.

Toeplitz 矩阵属于一个更大的**转对称** (persymmetric)[①] 矩阵类. 如果 $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ 是关于东北-西南对角线对称的, 即对所有 $i$ 和 $j$ 有 $b_{ij} = b_{n-j+1,n-i+1}$, 则称它是转对称的. 这个定义等价于要求 $\boldsymbol{B} = \boldsymbol{E}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{E}$, 其中 $\boldsymbol{E} = [e_n, \cdots, e_1] = I_n(:, n:-1:1)$ 是 $n \times n$ 的交换矩阵, 例如

$$\boldsymbol{E} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

容易验证: (a) Toeplitz 是转对称的. (b) 非奇异的 Toeplitz 矩阵的逆也是转对称的. 本节中我们将说明如何细致地利用 (b) 使能在 $O(n^2)$ 个 flop 内求解 Toeplitz 方程组. 我们主要讨论 $\boldsymbol{T}$ 是对称且正定的重要情形. 对于非对称的 Toeplitz 方程组及它与循环矩阵的联系, 还有离散的 Fourier 变换我们只做简单讨论.

### 4.7.1 三个问题

假定有标量 $r_1, \cdots, r_n$ 使得对于 $k = 1:n$, 矩阵

$$\boldsymbol{T}_k = \begin{bmatrix} 1 & r_1 & \cdots & r_{k-2} & r_{k-1} \\ r_1 & 1 & \ddots & & r_{k-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{k-2} & & \ddots & \ddots & r_1 \\ r_{k-1} & r_{k-2} & \cdots & r_1 & 1 \end{bmatrix}$$

是正定的 (不失一般性, 这里的对角元素是已单位化的). 本节介绍三个算法:

- 求解 Yule-Walker 问题 $\boldsymbol{T}_n\boldsymbol{y} = -[r_1, \cdots, r_n]^{\mathrm{T}}$ 的 Durbin 算法
- 求解一般问题 $\boldsymbol{T}_n\boldsymbol{x} = \boldsymbol{b}$ 的 Levinson 算法
- 计算 $\boldsymbol{B} = \boldsymbol{T}_n^{-1}$ 的 Trend 算法

为推出这些算法, 我们用 $\boldsymbol{E}_k$ 表示 $k \times k$ 的交换矩阵, 即 $\boldsymbol{E}_k = I_k(:, k:-1:1)$.

### 4.7.2 求解 Yule-Walker 方程

我们先给出求解 Yule-Walker 方程的 Durbin 算法, 该方程与一些线性预测问题密切相关. 假定 $k$ 满足 $1 \leqslant k \leqslant n-1$, 且我们已求解了 $k$ 阶 Yule-Walker 方程组 $\boldsymbol{T}_k\boldsymbol{y} = -\boldsymbol{r} = -(r_1, \cdots, r_k)^{\mathrm{T}}$. 下面给出如何在 $O(k)$ 个 flop 之内求解 $(k+1)$ 阶 Yule-Walker 方程组:

$$\begin{bmatrix} \boldsymbol{T}_k & \boldsymbol{E}_k\boldsymbol{r} \\ \boldsymbol{r}^{\mathrm{T}}\boldsymbol{E}_k & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{z} \\ \alpha \end{bmatrix} = -\begin{bmatrix} \boldsymbol{r} \\ r_{k+1} \end{bmatrix}.$$

---

① persymmetric 和 symmetric 几乎是一样的东西, 只不过一个是关于从左上角到左下角的对角线对称, 另一个是关于从右上角到左下角的对角线对称, 这里译成 “转对称”, 意思是将矩阵旋转 90 度就是原来的对称. —— 译者注

首先应注意

$$z = T_k^{-1}(-r - \alpha E_k r) = y - \alpha T_k^{-1} E_k r,$$
$$\alpha = -r_{k+1} - r^{\mathrm{T}} E_k z.$$

由于 $T_k^{-1}$ 是转对称的, $T_k^{-1} E_k = E_k T_k^{-1}$, 于是

$$z = y - \alpha E_k T_k^{-1} r = y + \alpha E_k y.$$

将此代入上面的 $\alpha$ 表达式, 有

$$\alpha = -r_{k+1} - r^{\mathrm{T}} E_k (y + \alpha E_k y) = -(r_{k+1} + r^{\mathrm{T}} E_k y)/(1 + r^{\mathrm{T}} y).$$

上式中的分母是大于零的, 这是因为 $T_{k+1}$ 是正定的, 而且

$$\begin{bmatrix} I & E_k y \\ 0 & 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} T_k & E_k r \\ r^{\mathrm{T}} E_k & 1 \end{bmatrix} \begin{bmatrix} I & E_k y \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} T_k & 0 \\ 0 & 1 + r^{\mathrm{T}} y \end{bmatrix}.$$

这就完成了 Durbin(1960) 给出的算法的第 $k$ 步描述. 对 $k = 1 : n$ 求解 Yule-Walker 方程组

$$T_k y^{(k)} = -r^{(k)} = -[r_1, \cdots, r_k]^{\mathrm{T}},$$

就得到如下算法:

$$\begin{aligned}
& y^{(1)} = -r_1 \\
& \textbf{for } k = 1 : n-1 \\
& \qquad \beta_k = 1 + [r^{(k)}]^{\mathrm{T}} y^{(k)} \\
& \qquad \alpha_k = -(r_{k+1} + r^{(k)^{\mathrm{T}}} E_k y^{(k)})/\beta_k \\
& \qquad z^{(k)} = y^{(k)} + \alpha_k E_k y^{(k)} \\
& \qquad y^{(k+1)} = \begin{bmatrix} z^{(k)} \\ \alpha_k \end{bmatrix} \\
& \textbf{end}
\end{aligned} \tag{4.7.1}$$

本算法需 $3n^2$ 个 flop 来产生 $y = y^{(n)}$. 如果充分利用一些上面的表达式, 可以进一步减少工作量:

$$\begin{aligned}
\beta_k &= 1 + [r^{(k)}]^{\mathrm{T}} y^{(k)} \\
&= 1 + \begin{bmatrix} (r^{(k-1)})^{\mathrm{T}} & r_k \end{bmatrix} \begin{bmatrix} y^{(k-1)} + \alpha_{k-1} E_{k-1} y^{(k-1)} \\ \alpha_{k-1} \end{bmatrix} \\
&= (1 + [r^{(k-1)}]^{\mathrm{T}} y^{(k-1)}) + \alpha_{k-1}([r^{(k-1)}]^{\mathrm{T}} E_{k-1} y^{(k-1)} + r_k) \\
&= \beta_{k-1} + \alpha_{k-1}(-\beta_{k-1}\alpha_{k-1}) \\
&= (1 - \alpha_{k-1}^2)\beta_{k-1}.
\end{aligned}$$

应用这个递推式得到下面的算法.

**算法 4.7.1 (Durbin)** 给定实数 $1 = r_0, r_1, \cdots, r_n$ 使得 $T = (r_{|i-j|}) \in \mathbb{R}^{n \times n}$ 是正定的, 本算法计算满足 $Ty = -(r_1, \cdots, r_n)^{\mathrm{T}}$ 的 $y \in \mathbb{R}^n$.

$y(1) = -r(1); \beta = 1; \alpha = -r(1)$
**for** $k = 1 : n-1$
  $\beta = (1-\alpha^2)\beta$
  $\alpha = -(r(k+1) + r(k:-1:1)^{\mathrm{T}} y(1:k))/\beta$
  $z(1:k) = y(1:k) + \alpha y(k:-1:1)$
  $y(1:k+1) = \begin{bmatrix} z(1:k) \\ \alpha \end{bmatrix}$
**end**

本算法需 $2n^2$ 个 flop. 为清晰起见, 我们用了一个辅助向量 $z$, 实际上它可以省去.

**例 4.7.1** 假定我们用算法 4.7.1 求解 Yule-Walker 方程组

$$\begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.5 \\ 0.2 & 0.5 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = -\begin{bmatrix} 0.5 \\ 0.2 \\ 0.1 \end{bmatrix}.$$

循环完成后, 得到

$$\alpha = 1/15, \quad \beta = 3/4, \quad y = \begin{bmatrix} -8/15 \\ 1/15 \end{bmatrix}.$$

然后计算

$$\beta = (1-\alpha^2)\beta = 56/75$$
$$\alpha = -(r_3 + r_2 y_1 + r_1 y_2)/\beta = -1/28$$
$$z_1 = y_1 + \alpha y_2 = -225/420$$
$$z_2 = y_2 + \alpha y_1 = -36/420$$

得到最终解为 $\boldsymbol{y} = [-75, 12, -5]^{\mathrm{T}}/140$.

### 4.7.3 一般右端向量问题

增加少许计算量, 可以求解右端为任意向量的对称正定 Toeplitz 方程组. 假定对某个 $k$ 满足 $1 \leqslant k < n$, 我们已解出方程组

$$\boldsymbol{T}_k \boldsymbol{x} = \boldsymbol{b} = (b_1, \cdots, b_k)^{\mathrm{T}}, \tag{4.7.2}$$

现需求解

$$\begin{bmatrix} \boldsymbol{T}_k & \boldsymbol{E}_k \boldsymbol{r} \\ \boldsymbol{r}^{\mathrm{T}} \boldsymbol{E}_k & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \mu \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} \\ b_{k+1} \end{bmatrix}. \tag{4.7.3}$$

这里 $\boldsymbol{r} = (r_1, \cdots, r_k)^{\mathrm{T}}$ 同上. 假定 $k$ 阶 Yule-Walker 方程组 $\boldsymbol{T}_k \boldsymbol{y} = -\boldsymbol{r}$ 的解也已得到. 从 $\boldsymbol{T}_k \boldsymbol{v} + \mu \boldsymbol{E}_k \boldsymbol{r} = \boldsymbol{b}$ 可知

$$\boldsymbol{v} = \boldsymbol{T}_k^{-1}(\boldsymbol{b} - \mu \boldsymbol{E}_k \boldsymbol{r}) = \boldsymbol{x} - \mu \boldsymbol{T}_k^{-1} \boldsymbol{E}_k \boldsymbol{r} = \boldsymbol{x} + \mu \boldsymbol{E}_k \boldsymbol{y},$$

故

$$\begin{aligned}\mu &= b_{k+1} - \boldsymbol{r}^{\mathrm{T}}\boldsymbol{E}_k\boldsymbol{v} \\ &= b_{k+1} - \boldsymbol{r}^{\mathrm{T}}\boldsymbol{E}_k\boldsymbol{x} - \mu\boldsymbol{r}^{\mathrm{T}}\boldsymbol{y} \\ &= (b_{k+1} - \boldsymbol{r}^{\mathrm{T}}\boldsymbol{E}_k\boldsymbol{x})/(1+\boldsymbol{r}^{\mathrm{T}}\boldsymbol{y}).\end{aligned}$$

这样, 我们可在 $O(k)$ 个 flop 内完成 (4.7.2) 到 (4.7.3) 的转换.

总的来说, 我们可以通过对 $k=1:n$ 并行地求解 $\boldsymbol{T}_k\boldsymbol{x}^{(k)} = \boldsymbol{b}^{(k)} = (b_1,\cdots,b_k)^{\mathrm{T}}$ 和 $\boldsymbol{T}_k\boldsymbol{y}^{(k)} = -\boldsymbol{r}^{(k)} = (r_1,\cdots,r_k)^{\mathrm{T}}$ 来解出 $\boldsymbol{T}_n\boldsymbol{x} = \boldsymbol{b}$. 这就是下述算法的要点.

**算法 4.7.2 (Levinson)**　给定 $\boldsymbol{b}\in\mathbb{R}^n$ 和实数 $1=r_0,r_1,\cdots,r_n$ 使得 $\boldsymbol{T}=(r_{|i-j|})\in\mathbb{R}^{n\times n}$ 是正定的, 本算法计算满足 $\boldsymbol{T}\boldsymbol{x}=\boldsymbol{b}$ 的 $\boldsymbol{x}\in\mathbb{R}^n$.

$y(1) = -r(1); x(1) = b(1); \beta = 1 : \alpha = -r(1)$

**for** $k = 1 : n-1$

$\quad\beta = (1-\alpha^2)\beta; \mu = (b(k+1) - r(1:k)^{\mathrm{T}}x(k:-1:1))/\beta$

$\quad v(1:k) = x(1:k) + \mu y(k:-1:1)$

$\quad x(1:k+1) = \begin{bmatrix} v(1:k) \\ \mu \end{bmatrix}$

$\quad$**if** $k < n-1$

$\qquad\alpha = (-r(k+1) + r(1:k)^{\mathrm{T}}y(k:-1:1))/\beta$

$\qquad z(1:k) = y(1:k) + \alpha y(k:-1:1)$

$\qquad y(1:k+1) = \begin{bmatrix} z(1:k) \\ \alpha \end{bmatrix}$

$\quad$**end**

**end**

本算法需 $4n^2$ 个 flop. 向量 $\boldsymbol{z}$ 和 $\boldsymbol{v}$ 是为使算法清晰而引入的, 实际编程时可避免.

**例 4.7.2**　假设要用以上算法求解对称正定 Toeplitz 方程组

$$\begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.5 \\ 0.2 & 0.5 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = -\begin{bmatrix} 4 \\ -1 \\ 3 \end{bmatrix}.$$

循环执行完后, 得到

$$\alpha = 1/15,\quad \beta = 3/4,\quad \boldsymbol{y} = \begin{bmatrix} -8/15 \\ 1/15 \end{bmatrix},\quad \boldsymbol{x} = \begin{bmatrix} 6 \\ -4 \end{bmatrix}.$$

然后计算

$$\beta = (1-\alpha^2)\beta = 56/75,\quad \mu = (b_3 - r_1x_2 - r_2x_1)/\beta = 285/56,$$

$$v_1 = x_1 + \mu y_2 = 355/56,\quad v_2 = x_2 + \mu y_1 = -376/56$$

得到最终解为: $\boldsymbol{x} = (355, -376, 285)^{\mathrm{T}}/56$.

### 4.7.4 求逆

对称正定的 Toeplitz 矩阵 $T_n$ 的最令人吃惊的性质之一是它的逆可在 $O(n^2)$ 个 flop 内算出. 为推出这个求解算法, 将 $T_n^{-1}$ 作如下划分:

$$T_n^{-1} = \begin{bmatrix} A & Er \\ r^{\mathrm{T}}E & 1 \end{bmatrix}^{-1} = \begin{bmatrix} B & v \\ v^{\mathrm{T}} & \gamma \end{bmatrix} \tag{4.7.4}$$

其中 $A = T_{n-1}, E = E_{n-1}, r = (r_1, \cdots, r_{n-1})^{\mathrm{T}}$. 由等式

$$\begin{bmatrix} A & Er \\ r^{\mathrm{T}}E & 1 \end{bmatrix} \begin{bmatrix} v \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

可知 $Av = -\gamma Er = -\gamma E(r_1, \cdots, r_{n-1})^{\mathrm{T}}$ 和 $\gamma = 1 - r^{\mathrm{T}}Ev$. 如果 $y$ 是 $(n-1)$ 阶 Yule-Walker 方程组 $Ay = -r$ 的解, 则由这些表达式可推出

$$\gamma = 1/(1 + r^{\mathrm{T}}y), \quad v = \gamma Ey.$$

这样就得到了 $T_{n-1}^{-1}$ 的最后一行和最后一列.

剩下的工作是建立求解 (4.7.4) 中子矩阵 $B$ 的元素的计算公式. 因为 $AB + Erv^{\mathrm{T}} = I_{n-1}$, 故有

$$B = A^{-1} - (A^{-1}Er)v^{\mathrm{T}} = A^{-1} + \frac{vv^{\mathrm{T}}}{\gamma}.$$

由于 $A = T_{n-1}$ 是非奇异的 Teoplitz 矩阵, 其逆是转对称的. 于是

$$\begin{aligned} b_{ij} &= (A^{-1})_{ij} + \frac{v_i v_j}{\gamma} \\ &= (A^{-1})_{n-j,n-i} + \frac{v_i v_j}{\gamma} \\ &= b_{n-j,n-i} - \frac{v_{n-j}v_{n-i}}{\gamma} + \frac{v_i v_j}{\gamma} \\ &= b_{n-j,n-i} + \frac{1}{\gamma}(v_i v_j - v_{n-j}v_{n-i}). \end{aligned} \tag{4.7.5}$$

这个公式说明尽管 $B$ 不是转对称的, 但我们可以通过所求 $b_{ij}$ 的关于东北–西南轴的反射点来计算它. 连同 $A^{-1}$ 是转对称的事实可使我们 “从外到里” 来确定 $B$.

由于描绘运算的次序相当麻烦, 我们以图示的形式预览一下前面提出的方案. 为此, 假定 $T_n^{-1}$ 的最后一行和最后一列已知

$$T_n^{-1} = \begin{bmatrix} u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ k & k & k & k & k & k \end{bmatrix}.$$

这里 $u$ 和 $k$ 分别代表未知和已知元素, $n=6$. 交替利用 $\boldsymbol{T}_n^{-1}$ 的转对称性和递推式 (4.7.5), 可求出 $\boldsymbol{T}_{n-1}^{-1}$ 的 $(n-1)\times(n-1)$ 的顺序子块 $\boldsymbol{B}$ 如下:

$$\xrightarrow{\text{转对称}}\begin{bmatrix} k & k & k & k & k & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & k & k & k & k & k \end{bmatrix} \xrightarrow{(4.7.5)} \begin{bmatrix} k & k & k & k & k & k \\ k & u & u & u & k & k \\ k & u & u & u & k & k \\ k & u & u & u & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{bmatrix}$$

$$\xrightarrow{\text{转对称}}\begin{bmatrix} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & u & u & k & k \\ k & k & u & u & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{bmatrix} \xrightarrow{(4.7.5)} \begin{bmatrix} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & u & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{bmatrix}$$

$$\xrightarrow{\text{转对称}}\begin{bmatrix} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{bmatrix}.$$

当然, 当求解一个既对称又转对称的矩阵 (如 $T_n^{-1}$) 时, 只需求解矩阵的一个 "上锲形" 即可. 即

$$\begin{matrix} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \\ & & \times & \times & & \end{matrix} \qquad (n=6).$$

由以上分析, 可以给出完整的算法.

**算法 4.7.3 (Trench)** 给定实数 $1=r_0,r_1,\cdots,r_n$ 使得 $\boldsymbol{T}=(r_{|i-j|})\in\mathbb{R}^{n\times n}$ 是对称正定的, 本算法计算 $\boldsymbol{B}=\boldsymbol{T}_n^{-1}$, 只计算满足 $i\leqslant j$ 和 $i+j\leqslant n+1$ 的 $b_{ij}$.

用算法 4.7.1 求解 $T_{n-1}y=-(r_1,\cdots,r_{n-1})^{\mathrm{T}}$.
$\gamma=1/(1+r(1:n-1)^{\mathrm{T}}y(1:n-1))$
$v(1:n-1)=\gamma y(n-1:-1:1)$
$B(1,1)=\gamma$
$B(1,2:n)=v(n-1:-1:1)^{\mathrm{T}}$
**for** $i=2:\mathrm{floor}((n-1)/2)+1$
  **for** $j=i:n-i+1$
    $B(i,j)=B(i-1,j-1)+$

$$(v(n+1-j)v(n+1-i)-v(i-1)v(j-1))/\gamma$$

**end**

**end**

本算法需 $13n^2/4$ 个 flop.

**例 4.7.3** 假设用上述算法求对称正定 Toeplitz 矩阵

$$\begin{bmatrix} 1 & 0.5 & 0.2 \\ 0.5 & 1 & 0.5 \\ 0.2 & 0.5 & 1 \end{bmatrix}$$

的逆矩阵 $\boldsymbol{B}$, 则我们得到 $\gamma = 75/56$, $b_{11} = 75/56$, $b_{12} = -5/7$, $b_{13} = 5/56$, $b_{22} = 12/7$.

### 4.7.5 稳定性问题

Cybenko (1978) 对上述诸算法做了误差分析, 我们将简要介绍他的结果.

关键的量是 (4.7.1) 中的 $\alpha_k$. 精确运算时, 它应满足 $|\alpha_k| < 1$, 且可被用来对 $\|\boldsymbol{T}^{-1}\|_1$ 限界:

$$\max\left\{\frac{1}{\prod_{j=1}^{n-1}(1-\alpha_j^2)}, \frac{1}{\prod_{j=1}^{n-1}(1-\alpha_j)}\right\} \leqslant \|\boldsymbol{T}_n^{-1}\| \leqslant \prod_{j=1}^{n-1}\frac{1+|\alpha_j|}{1-|\alpha_j|}. \tag{4.7.6}$$

而且, 只要所有的 $\alpha_k$ 都非负, Yule-Walker 方程组 $\boldsymbol{T}_n\boldsymbol{y} = -r(1:n)$ 之解就满足:

$$\|\boldsymbol{y}\|_1 = \left(\prod_{k=1}^{n}(1+\alpha_k)\right) - 1. \tag{4.7.7}$$

如果 $\hat{\boldsymbol{x}}$ 是 Yule-Walker 方程的 Durbin 算法之计算解, 则 $\boldsymbol{r}_D = \boldsymbol{T}_n\hat{\boldsymbol{x}} + \boldsymbol{r}$ 可被限界如下:

$$\|\boldsymbol{r}_D\| \approx \mathbf{u}\prod_{k=1}^{n}(1+|\hat{\alpha}_k|),$$

其中 $\hat{\alpha}_k$ 是 $\alpha_k$ 的计算解. 通过比较, 因为每个 $|r_i|$ 都是以 1 为界, 故 $\|\boldsymbol{r}_C\| \approx \mathbf{u}\|\boldsymbol{y}\|_1$, 其中 $\boldsymbol{r}_C$ 是用 Cholesky 法求得的计算解的剩余. 注意, 如 (4.7.7) 成立, 则这两个剩余的大小差不多. 实际的数据表明即使一部分 $\alpha_k$ 小于零时的情形也是如此. 对于 Levinson 算法的数值特性我们可得到类似的结论.

至于 Trench 算法, 可以证明 $\boldsymbol{T}_n^{-1}$ 的计算逆 $\hat{\boldsymbol{B}}$ 满足

$$\frac{\|\boldsymbol{T}_n^{-1} - \hat{B}\|_1}{\|\boldsymbol{T}_n^{-1}\|_1} \approx \mathbf{u}\prod_{k=1}^{n}\frac{1+|\hat{\alpha}_k|}{1-|\hat{\alpha}_k|}.$$

根据 (4.7.7) 可以看出右端是 $\mathbf{u}\|\boldsymbol{T}_n^{-1}\|$ 的近似上界, 而 $\mathbf{u}\|\boldsymbol{T}_n^{-1}\|$ 之大小大约是用 Cholesky 法计算 $\boldsymbol{T}_n^{-1}$ 时的相对误差.

### 4.7.6 非对称的情况

对非对称情况可推出类似的递推算法. 假定已给数 $r_1,\cdots,r_{n-1}$ 和 $p_1,\cdots,p_{n-1}$, 要求解形如

$$\begin{bmatrix} 1 & r_1 & r_2 & r_3 & r_4 \\ p_1 & 1 & r_1 & r_2 & r_3 \\ p_2 & p_1 & 1 & r_1 & r_2 \\ p_3 & p_2 & p_1 & 1 & r_1 \\ p_4 & p_3 & p_2 & p_1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} \quad (n=5).$$

的线性方程组 $\boldsymbol{T}x=b$.

下面的推导过程要求 $\boldsymbol{T}$ 的顺序主子阵 $\boldsymbol{T}_k=\boldsymbol{T}(1:k,1:k), k=1:n$ 是非奇异的. 使用与上相同的记号, 可以证明: 如果已有 $k\times k$ 方程组的解:

$$\begin{aligned} \boldsymbol{T}_k^{\mathrm{T}}\boldsymbol{y} &= -\boldsymbol{r} = -[r_1 \quad r_2 \quad \cdots \quad r_k]^{\mathrm{T}}, \\ \boldsymbol{T}_k\boldsymbol{\omega} &= -\boldsymbol{p} = -[p_1 \quad p_2 \quad \cdots \quad p_k]^{\mathrm{T}}, \\ \boldsymbol{T}_k\boldsymbol{x} &= \boldsymbol{b} \quad = [b_1 \quad b_2 \quad \cdots \quad b_k]^{\mathrm{T}}, \end{aligned} \tag{4.7.8}$$

则可在 $O(k)$ 个 flop 内得到下面方程组的解:

$$\begin{aligned} &\begin{bmatrix} \boldsymbol{T}_k & \boldsymbol{E}_k\boldsymbol{r} \\ \boldsymbol{p}^{\mathrm{T}}\boldsymbol{E}_k & 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} z \\ \alpha \end{bmatrix} = -\begin{bmatrix} r \\ r_{k+1} \end{bmatrix} \\ &\begin{bmatrix} \boldsymbol{T}_k & \boldsymbol{E}_k\boldsymbol{r} \\ \boldsymbol{p}^{\mathrm{T}}\boldsymbol{E}_k & 1 \end{bmatrix} \begin{bmatrix} u \\ \nu \end{bmatrix} = -\begin{bmatrix} p \\ p_{k+1} \end{bmatrix} \\ &\begin{bmatrix} \boldsymbol{T}_k & \boldsymbol{E}_k\boldsymbol{r} \\ \boldsymbol{p}^{\mathrm{T}}\boldsymbol{E}_k & 1 \end{bmatrix} \begin{bmatrix} v \\ \mu \end{bmatrix} = \begin{bmatrix} b \\ b_{k+1} \end{bmatrix} \end{aligned} \tag{4.7.9}$$

这意味着理论上能在 $O(n^2)$ 个 flop 内求解一个非对称 Toeplitz 方程组. 然而, 除非矩阵 $\boldsymbol{T}_k=T(1:k,1:k)$ 有很好的条件数, 否则算法的稳定性得不到保证.

### 4.7.7 循环方程组

有一类非常重要的 Toeplitz 矩阵是**循环矩阵**. 以下是一个例子:

$$\boldsymbol{C}(v) = \begin{bmatrix} v_0 & v_4 & v_3 & v_2 & v_1 \\ v_1 & v_0 & v_4 & v_3 & v_2 \\ v_2 & v_1 & v_0 & v_4 & v_3 \\ v_3 & v_2 & v_1 & v_0 & v_4 \\ v_4 & v_3 & v_2 & v_1 & v_0 \end{bmatrix}.$$

注意到循环矩阵的每一列都是将其前一列向下移动一个位置得到的. 如果定义下移置换矩阵为

$$
\boldsymbol{S}_n = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (n=5),
$$

且 $\boldsymbol{v} = [v_0 \ v_1 \cdots \ v_{n-1}]^{\mathrm{T}}$, 则 $\boldsymbol{C}(\boldsymbol{v}) = [\boldsymbol{v}, \boldsymbol{S}_n\boldsymbol{v}, \boldsymbol{S}_n^2\boldsymbol{v}, \cdots, \boldsymbol{S}_n^{n-1}\boldsymbol{v}]$.

循环矩阵、Toeplitz 矩阵和 DFT 之间有很重要的联系. 首先, 可以证明

$$
\boldsymbol{C}(\boldsymbol{v}) = \boldsymbol{F}_n^{-1}\mathrm{diag}(\boldsymbol{F}_n\boldsymbol{v})\boldsymbol{F}_n. \tag{4.7.10}
$$

这意味着形如 $\boldsymbol{y} = \boldsymbol{C}(\boldsymbol{v})\boldsymbol{x}$ 的积能够以 “FFT 速度” 求解:

$$
\begin{aligned}
\widetilde{\boldsymbol{x}} &= \boldsymbol{F}_n\boldsymbol{x}, \\
\widetilde{\boldsymbol{v}} &= \boldsymbol{F}_n\boldsymbol{v}, \\
\boldsymbol{z} &= \widetilde{\boldsymbol{v}}.*\widetilde{\boldsymbol{x}}, \\
\boldsymbol{y} &= \boldsymbol{F}_n^{-1}\boldsymbol{z}.
\end{aligned}
$$

换句话说, 三次 DFT 运算和一次向量乘法足以求解一个循环矩阵与一个向量的乘积, 这种形式的积称为*卷积*, 它遍布于信号处理及其他领域.

Toeplitz 矩阵与向量的乘积也可快速计算出. 关键思想是任何 Toeplitz 矩阵都可被嵌入到一个循环矩阵中. 例如

$$
\boldsymbol{T} = \begin{bmatrix} 5 & 2 & 7 \\ 4 & 5 & 2 \\ 9 & 4 & 5 \end{bmatrix},
$$

它是

$$
\boldsymbol{C} = \left[\begin{array}{ccc|cc} 5 & 2 & 7 & 9 & 4 \\ 4 & 5 & 2 & 7 & 9 \\ 9 & 4 & 5 & 2 & 7 \\ \hline 7 & 9 & 4 & 5 & 2 \\ 2 & 7 & 9 & 4 & 5 \end{array}\right]
$$

的顺序 $3\times 3$ 子矩阵.

一般地, 如 $\boldsymbol{T} = (t_{ij})$ 是 $n\times n$ 的 Toeplitz 矩阵, 则 $\boldsymbol{T} = C(1:n, 1:n)$, 其中 $\boldsymbol{C} \in \mathbb{R}^{(2n-1)\times(2n-1)}$ 是循环矩阵, 且

$$
C(:,1) = \begin{bmatrix} T(1:n,1) \\ T(1,n:-1:2)^{\mathrm{T}} \end{bmatrix}.
$$

注意到, 如果 $\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}$ 且 $x(n+1:2n-1) = 0$, 则 $y(1:n) = \boldsymbol{T}x(1:n)$, 这说明 Toeplitz 向量乘积也能以 “FFT 速度” 计算出.

## 习 题

**4.7.1** 对任意 $v \in \mathbb{R}^n$ 定义向量 $v_+ = (v + E_n v)/2$ 和 $v_- = (v - E_n v)/2$, 假定 $A \in \mathbb{R}^{n\times n}$ 是对称且转对称的, 证明如果 $Ax = b$, 则 $Ax_+ = b_+$ 和 $Ax_- = b_-$.

**4.7.2** 令 $U \in \mathbb{R}^{n\times n}$ 是单位上三角形矩阵且有性质: $U(1:k-1,k) = E_{k-1}y^{(k-1)}$, 其中 $y^{(k)}$ 由 (4.7.1) 定义. 证明

$$U^{\mathrm{T}}T_nU = \mathrm{diag}(1, \beta_1, \cdots, \beta_{n-1}).$$

**4.7.3** 假设 $z \in \mathbb{R}^n$ 且 $S \in \mathbb{R}^{n\times n}$ 是正交矩阵, 证明: 如果

$$X = [z, Sz, \cdots, S^{n-1}z],$$

则 $X^{\mathrm{T}}X$ 是 Toeplitz 矩阵.

**4.7.4** 考虑 $n\times n$ 阶三对角对称正定 Toeplitz 矩阵的 $\mathrm{LDL^T}$ 分解, 证明当 $n\to\infty$ 时, $d_n$ 和 $l_{n,n-1}$ 收敛.

**4.7.5** 证明两个下三角形 Toeplitz 矩阵的乘积仍是 Toeplitz 矩阵.

**4.7.6** 给出一个算法求出 $\mu \in \mathbb{R}$ 使得

$$T_n + \mu(e_ne_1^{\mathrm{T}} + e_1e_n^{\mathrm{T}})$$

是奇异的. 假定 $T_n = (r_{|i-j|})$ 是正定的, 其中 $r_0 = 1$.

**4.7.7** 改写算法 4.7.2, 要求不使用辅助向量 $v$ 和 $z$.

**4.7.8** 给出一个计算 $k_\infty(T_k), k = 1:n$ 的算法.

**4.7.9** 假定 $A_1, A_2, A_3, A_4$ 是 $m\times m$ 矩阵且

$$A = \begin{bmatrix} A_0 & A_1 & A_2 & A_3 \\ A_3 & A_0 & A_1 & A_2 \\ A_2 & A_3 & A_0 & A_1 \\ A_1 & A_2 & A_3 & A_0 \end{bmatrix},$$

证明存在一个置换矩阵 $\Pi$ 满足 $\Pi^{\mathrm{T}}A\Pi = C = (C_{ij})$, 其中每个 $C_{ij}$ 是 $4\times 4$ 循环矩阵.

**4.7.10** 对于一个含有 $p\times p$ 个块, 每块大小为 $m\times m$ 的分块矩阵 $A$, 如果存在 $A_{-p+1},\cdots,A_{-1},A_0A_1,\cdots,A_{p-1} \in \mathbb{R}^{m\times m}$ 满足 $A_{ij} = A_{i-j}$, 即

$$A = \begin{bmatrix} A_0 & A_1 & A_2 & A_3 \\ A_{-1} & A_0 & A_1 & A_2 \\ A_{-2} & A_{-1} & A_0 & A_1 \\ A_{-3} & A_{-2} & A_{-1} & A_0 \end{bmatrix},$$

则 $A$ 是分块 Toeplitz 矩阵.

(a) 证明存在置换矩阵 $\Pi$ 满足

$$\Pi^{\mathrm{T}}A\Pi = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ T_{21} & T_{22} & & \vdots \\ \vdots & & \ddots & \\ T_{m1} & \cdots & & T_{mm} \end{bmatrix}$$

其中每个 $T_{ij}$ 是 $p\times p$ 的 Toeplitz 矩阵, 每个 $T_{ij}$ 是由 $A_k$ 的 $(i,j)$ 元素组成的.

(b) 如 $\boldsymbol{A}_k = \boldsymbol{A}_{-k}, k = 1:p-1$, 则 $\boldsymbol{T}_{ij}$ 是什么样的?

**4.7.11** 如给定 (4.7.8) 的解, 那么如何计算 (4.7.9) 的解? 假设所涉及的所有矩阵均为非奇异的. 假设 $\boldsymbol{T}$ 的所有顺序主子矩阵均为非奇异的, 进一步设计一个快速算法求解非对称 Toeplitz 方程组 $\boldsymbol{T}\boldsymbol{x} = \boldsymbol{b}$.

**4.7.12** 当 $H(n:-1:1\ :)$ 是 Toeplitz 矩阵时矩阵 $\boldsymbol{H} \in \mathbb{R}^{n\times n}$ 称为 Hankel 矩阵. 证明: 如果定义 $\boldsymbol{A}$ 为

$$a_{ij} = \int_a^b \cos(k\theta)\cos(j\theta)\mathrm{d}\theta,$$

则 $\boldsymbol{A}$ 是 Hankel 矩阵与 Toeplitz 矩阵的和. 提示: 利用等式 $\cos(u+v) = \cos(u)\cos(v) - \sin(u)\sin(v)$.

**4.7.13** 证明 $\boldsymbol{F}_n\boldsymbol{C}(v) = \mathrm{diag}(\boldsymbol{F}_n v)\boldsymbol{F}_n$.

**4.7.14** 证明能够将一个对称 Toeplitz 矩阵嵌入到一个对称循环矩阵中.

**4.7.15** 考虑 (4.7.1) 中的 $k$ 阶 Yule-Walker 方程组 $\boldsymbol{T}_k\boldsymbol{y}^{(k)} = -\boldsymbol{r}^{(k)}$

$$\boldsymbol{T}_k \begin{bmatrix} y_{k1} \\ \vdots \\ y_{kk} \end{bmatrix} = -\begin{bmatrix} r_1 \\ \vdots \\ r_k \end{bmatrix},$$

证明: 如果

$$\boldsymbol{L} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ y_{11} & 1 & 0 & 0 & \cdots & 0 \\ y_{22} & y_{21} & 1 & 0 & \cdots & 0 \\ y_{33} & y_{32} & y_{31} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{n-1,n-1} & y_{n-1,n-2} & y_{n-1,n-3} & \cdots & y_{n-1,1} & 1 \end{bmatrix},$$

则 $\boldsymbol{L}\boldsymbol{T}_n\boldsymbol{L}^{\mathrm{T}} = \mathrm{diag}(1, \beta_1, \cdots, \beta_{n-1})$, 其中 $\beta_k = 1 + \boldsymbol{r}^{(k)\mathrm{T}}\boldsymbol{y}^{(k)}$. 于是 Durbin 算法可被看作计算 $\boldsymbol{T}_n^{-1}$ 的 $\mathrm{LDL^T}$ 分解的快速方法.

## 本节注释与参考文献

任何一个无目标地搜索大量的 Toeplitz 方法的文献的人应先注意到下文对稳定性的讨论:

J. R. Bunch (1985). "Stability of Methods for Solving Toeplitz Systems of Equations," *SIAM J. Sci. Stat. Comp.* 6, 349–364.

事实上, 在一般的 "快速算法" 范围内, 不稳定的 Toeplitz 技术大量存在, 必须小心处理. 参阅:

G. Cybenko (1978), "Error Analysis of Some Signal Processing Algorithms," Ph. D. thesis, Princeton University.

G. Cybenko (1980), "The Numerical Stability of the Levinson-Durbin Algorithm for Toeplitz Systems of Equations," *SIAM J. Sci. and Stat. Comp.* 1, 303–319.

E. Linzer (1992). "On the Stability of Solution Methods for Band Toeplitz Systems," *Lin. Alg. and Its Application* 170, 1–32.

J. M. Varah (1994). "Backward Error Estimates for Toeplitz Systems," *SIAM J. Matrix Anal. Appl.* 15, 408–417.

A. W. Bojanczyk, R. P. Brent, F. R. de Hoog, and D. R. Sweet (1995). "On the Stability of the Bareiss and Related Toeplitz Factorization Algorithms," *SIAM J. Matrix Anal. Appl.* 16, 40–57.

M. Stewart and P. Van Dooren (1996). "Stability Issues in the Factorization of Structured Matrices," *SIAM J. Matrix Anal. Appl.* 18, to appear.

本节给出的三个算法最早见于:

J. Durbin (1960). "The Fitting of Time Series Models," *Rev. Inst. Int. Stat.* 28, 233–243.

N. Levinson (1947). "The Weiner RMS Error Criterion in Filter Design and Prediction," *J. Math. Phys.* 25, 261–278.

W. F. Trench (1964). "An Algorithm for the Inversion of Finite Toeplitz Matrices," *J. SIAM* 12, 515–522.

非对称 Trench 算法的更详尽的描述见:

S. Zohar (1969). "Toeplitz Matrix Inversion: The Algorithm of W. F. Trench," *J. ACM* 16, 592–601.

快速地求解 Toeplitz 方程组吸引了无数人的注意力, 一些有趣的算法思想可在下述文献中查到:

G. Ammar and W. B. Gragg (1988). "Superfast Solution of Real Positive Definite Toeplitz Systems," *SIAM J. Matrix Anal. Appl.* 9, 61–76.

T. F. Chan and P. Hansen (1992). "A Look-Ahead Levinson Algorithm for Indefinite Toeplitz Systems," *SIAM J. Matrix Anal. Appl.* 13, 490–506.

D. R. Sweet (1993). "The Use of Pivoting to Improve the Numerical Performance of Algorithms for Toeplitz Matrices," *SIAM J. Matrix Anal. Appl.* 14, 468–493.

T. Kailath and J. Chun (1994). "Generalized Displacement Structure for Block-Toeplitz, Toeplitz-Block, and Toeplitz-Derived Matrices," *SIAMJ. Matrix Anal. Appl.* 15, 114–128.

T. Kailath and A. H. Sayed (1995). "Displacement Structure: Theory and Applications," *SIAM Review* 37, 297–386.

Toeplitz 矩阵的重要应用见于:

J. Makhoul (1975). "Linear Prediction: A Tutorial Review," *Proc. IEEE* 63(4), 561–580.

J. Markel and A. Gray (1976). *Linear Prediction of Speech*, Springer-Verlag, Berlin and New York.

A. V. Oppenheim (1978). *Applications of Digital Signal Processing*, Prentice-Hall, Englewood Cliffs.

Hankel 矩阵的反对角线元素是常数, 此类矩阵出现在许多重要的领域:

G. Heinig and P. Jankowski (1990). "Parallel and Superfast Algorithms for Hankel Systems of Equations," *Numer. Math.* 58, 109–127.

R. W. Freund and H. Zha (1993). "A Look-Ahead Algorithm for the Solution of General Hankel Systems," *Numer. Math.* 64, 295–322.

DFT、Toeplitz 矩阵以及循环矩阵之间的相互关系可见:

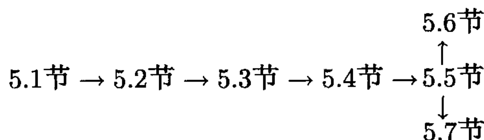
C. F. Van Loan (1992). *Computational Frameworks for the Fast Fourier Transform*, SIAM Publications, Philadelphia, PA.

# 第 5 章　正交化和最小二乘法

本章主要讨论超定方程组的最小二乘解, 即 $\|Ax-b\|_2$ 的最小化, 其中 $A\in\mathbb{R}^{n\times n}$. 解这个问题的最可靠的办法是通过正交变换将 $A$ 约化为各种典范型. Householder 反射和 Givens 旋转是这种方法的核心, 在本章的开始先对这两种重要的变换进行讨论. 5.2 节将讨论如何计算分解 $A=QR$, 其中 $Q$ 是正交矩阵, $R$ 是上三角形矩阵. 这相当于寻找 $\text{ran}(A)$ 的一个正交基. 我们在 5.3 节阐明, QR 分解可用来求解满秩的最小二乘问题. 给出扰动理论之后, 我们将此方法与法方程组方法作一比较. 在 5.4 节和 5.5 节中, 我们考虑那些处理 $A$ 是秩亏损 (或近于亏损) 的困难情形的方法. 主要介绍了选列主的 QR 分解和 SVD 分解. 在 5.6 节中, 给出提高最小二乘法计算解精度所采用的几个步骤. 5.7 节讨论正方形方程组和欠定方程组.

**预备知识**

阅读本章需要掌握第 1~3 章和 4.1 节 ~4.3 节的知识. 本章各节间的关系为




补充文献包括 Lawson and Hanson(1974), Farebrother(1987), Björck(1996). 还可见 Stewart(1973), Hager(1988), Stewart and Sun(1990), Watkins(1991), Gill Murray, and Wright(1991), Higham(1996), Trefethen and Bau(1996), Demmel(1996). 本章用到的重要的 MATLAB函数有qr, svd, pinv, orth, rank及反斜杠运算符 \ . 以下是与 LAPACK 相关的例程.

| LAPACK: Householder/Givens 工具 | |
|---|---|
| _LARFG | 产生 Householder 矩阵 |
| _LARF | Householder 矩阵乘矩阵 |
| _LARFX | 低维数的 Householder 矩阵乘矩阵 |
| _LARFB | 分块 Householder 矩阵乘矩阵 |
| _LARFT | 计算 $I-VTV^{\mathrm{H}}$ 分块反射表示 |
| _LARTG | 产生平面旋转 |
| _LARGV | 产生平面旋转的向量 |

(续)

| LAPACK: Householder/Givens 工具 | |
|---|---|
| _LARTV | 平面旋转作用于向量对 |
| _LASR | 旋转序列作用于矩阵 |
| _CSROT | 实旋转乘复向量对 |
| _CROT | 复旋转 (实 c) 乘复向量对 |
| _CLACGV | 复旋转 (实 s) 乘复向量对 |

| LAPACK: 正交分解 | |
|---|---|
| _GEQRF | $A = QR$ |
| _GEQPF | $A\Pi = QR$ |
| _ORMQR | $Q$(分解型) 乘矩阵 (实型) |
| _UNMQR | $Q$(分解型) 乘矩阵 (复型) |
| _ORGQR | 产生 $Q$(实型) |
| _UNGQR | 产生 $Q$(复型) |
| _GERQF | $A = RQ$ =(上三角)(正交) |
| _GELQF | $A = QL$ =(正交)(下三角) |
| _GEQLF | $A = LQ$ =(下三角)(正交) |
| _TZRQF | $A = RQ$, 其中 $A$ 是上梯形的 |
| _GESVD | $A = U\Sigma V^{\mathrm{T}}$ |
| _BDSQR | 实双对角矩阵的 SVD |
| _GEBRD | 一般矩阵的双对角化 |
| _ORGBR | 产生正交变换 |
| _GBBRD | 带状矩阵的双对角化 |

| LAPACK: 最小二乘 | |
|---|---|
| _GELS | 满秩的 $\min\Vert AX - B\Vert_F$ 或 $\min\Vert A^{\mathrm{H}}X - B\Vert_F$ |
| _GELSS | $\min\Vert AX - B\Vert_F$ 的 SVD 解 |
| _GELSX | $\min\Vert AX - B\Vert_F$ 的完全正交分解解 |
| _GEEQU | 平衡一般矩阵减小条件数 |

# 5.1 Householder 矩阵和 Givens 矩阵

由前面的知识, 如果 $Q^{\mathrm{T}}Q = QQ^{\mathrm{T}} = I_n$, 则 $Q \in \mathbb{R}^{n\times n}$ **是正交的**. 正交矩阵在最小二乘法和特征值计算方面起着重要作用. 本节我们介绍两个关键的变换: Householder 反射和 Givens 旋转.

## 5.1.1 一个 2×2 矩阵的引例

为引出主题, 先分析在 $n=2$ 的层次上的几何旋转和几何反射. 一个 $2\times 2$ 正交矩阵 $Q$, 如果有形式

$$Q = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix},$$

则称之为旋转变换. 如果 $y = Q^{\mathrm{T}}x$, 则 $y$ 是通过将 $x$ 逆时针旋转 $\theta$ 角度得到的.

一个 $2\times 2$ 正交矩阵 $Q$, 如果有形式

$$Q=\begin{bmatrix}\cos(\theta) & \sin(\theta)\\ \sin(\theta) & -\cos(\theta)\end{bmatrix}$$

则称之为反射变换. 如果 $y=Q^{\mathrm{T}}x=Qx$, 则 $y$ 是将向量 $x$ 关于由

$$S=\mathrm{span}\left\{\begin{bmatrix}\cos(\theta/2)\\ \sin(\theta/2)\end{bmatrix}\right\}$$

所定义的直线作反射得到的. 反射变换和旋转变换在计算上很有吸引力, 因为它们易于构造且可通过适当选取旋转角度和反射平面而使向量中产生新的零元素.

**例 5.1.1** 假定 $x=[1,\sqrt{3}]^{\mathrm{T}}$, 如果令

$$Q=\begin{bmatrix}\cos(-60^\circ) & \sin(-60^\circ)\\ -\sin(-60^\circ) & \cos(-60^\circ)\end{bmatrix}=\begin{bmatrix}1/2 & -\sqrt{3}/2\\ \sqrt{3}/2 & 1/2\end{bmatrix},$$

则 $Q^{\mathrm{T}}x=[2,0]^{\mathrm{T}}$. 因此, $-60^\circ$ 的旋转使 $x$ 的第二个分量化为零. 如果

$$Q=\begin{bmatrix}\cos(60^\circ) & \sin(60^\circ)\\ \sin(60^\circ) & -\cos(60^\circ)\end{bmatrix}=\begin{bmatrix}\sqrt{3}/2 & 1/2\\ 1/2 & -\sqrt{3}/2\end{bmatrix},$$

则 $Q^{\mathrm{T}}x=[2,0]^{\mathrm{T}}$. 于是, 将 $x$ 对 $30^\circ$ 的直线做反射也使第二个分量化为零.

### 5.1.2 Householder 反射

设 $v\in\mathbb{R}^n$ 是非零向量. 形如

$$P=I-\frac{2}{v^{\mathrm{T}}v}vv^{\mathrm{T}} \tag{5.1.1}$$

的 $n\times n$ 矩阵 $P$ 称为 Householder **反射**(同义词: Householder 矩阵, Householder 变换). 向量 $v$ 称作 Householder 向量. 如果用 $P$ 去乘向量 $x$, 就得到向量关于超平面 $\mathrm{span}\{v\}^{\perp}$ 的反射. 容易看出, Householder 矩阵是对称正交的.

Householder 反射与我们在 3.2.1 节介绍的高斯变换有两点类似. 二者都是单位矩阵的秩 1 校正形式, 都可用来将一个向量的某些选定分量变为零. 具体地说, 假设给定 $0\neq x\in\mathbb{R}^n$, 欲使 $Px$ 为 $e_1=I_n(:,1)$ 的一个倍数. 注意, 由

$$Px=\left(I-\frac{2vv^{\mathrm{T}}}{v^{\mathrm{T}}v}\right)x=x-\frac{2v^{\mathrm{T}}x}{v^{\mathrm{T}}v}v$$

和 $Px\in\mathrm{span}\{e_1\}$ 意味着 $v\in\mathrm{span}\{x,e_1\}$. 令 $v=x+\alpha e_1$, 得到

$$\begin{aligned}v^{\mathrm{T}}x&=x^{\mathrm{T}}x+\alpha x_1,\\ v^{\mathrm{T}}v&=x^{\mathrm{T}}x+2\alpha x_1+\alpha^2.\end{aligned}$$

因此

$$Px=\left(1-2\frac{x^{\mathrm{T}}x+\alpha x_1}{x^{\mathrm{T}}x+2\alpha x_1+\alpha^2}\right)x-2\alpha\frac{v^{\mathrm{T}}x}{v^{\mathrm{T}}v}e_1.$$

为了使 $x$ 的系数为零, 令 $\alpha=\pm\|x\|_2$, 于是

$$\boldsymbol{v} = \boldsymbol{x} \pm \|\boldsymbol{x}\|_2 \boldsymbol{e}_1 \Rightarrow \boldsymbol{P}\boldsymbol{x} = \left(\boldsymbol{I} - \frac{2\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}}{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}}\right)\boldsymbol{x} = \mp\|\boldsymbol{x}\|_2 \boldsymbol{e}_1. \tag{5.1.2}$$

正是能这样简单地确定 $\boldsymbol{v}$, 使得 Householder 反射非常有用.

**例 5.1.2** 如果 $\boldsymbol{x} = [3,1,5,1]^{\mathrm{T}}, \boldsymbol{v} = [9,1,5,1]^{\mathrm{T}}$, 则

$$\boldsymbol{P} = \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}}{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}} = \frac{1}{54}\begin{bmatrix} -27 & -9 & -45 & -9 \\ -9 & 53 & -5 & -1 \\ -45 & -5 & 29 & -5 \\ -9 & -1 & -5 & 53 \end{bmatrix}$$

具有性质 $\boldsymbol{P}\boldsymbol{x} = [-6,0,0,0]^{\mathrm{T}}$.

### 5.1.3 计算 Householder 向量

有许多确定 Householder 矩阵即 Householder 向量的重要细节. 其中之一是关于选择 (5.1.2) 中 $\boldsymbol{v}$ 的符号. 关系式

$$v_1 = x_1 - \|\boldsymbol{x}\|_2$$

具有 $\boldsymbol{P}\boldsymbol{x}$ 是 $\boldsymbol{e}_1$ 的一个正倍数这一良好性质. 当 $\boldsymbol{x}$ 接近 $e_1$ 的一个正倍数时, 由于会出现严重的相消, 因此这种方法是危险的. 然而 Parlett(1971) 提出的公式

$$\boldsymbol{v} = x_1 - \|\boldsymbol{x}\|_2 = \frac{\boldsymbol{x}_1^2 - \|\boldsymbol{x}\|_2^2}{x_1 + \|\boldsymbol{x}\|_2} = \frac{-(x_2^2 + \cdots + x_n^2)}{x_1 + \|\boldsymbol{x}\|_2}$$

在 $x_1 > 0$ 的情况下能避免这个缺陷.

实际上, 方便的是将 Householder 向量规范化使得 $v(1) = 1$. 这就允许将 $v(2:n)$ 存储到 $\boldsymbol{x}$ 已化为零的位置, 即 $x(2:n)$. 我们将 $v(2:n)$ 称为 Householder 向量的基本部分. 回忆 $\beta = 2/\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}$, 且令 length($\boldsymbol{x}$) 表示向量的维数, 我们得到下面的算法.

**算法 5.1.1 (Householder 向量)** 给定 $\boldsymbol{x} \in \mathbb{R}^n$, 本函数计算满足 $v(1) = 1$ 的 $\boldsymbol{v} \in \mathbb{R}^n$ 和 $\beta \in \mathbb{R}$, 其中 $\beta$ 使得 $\boldsymbol{P} = \boldsymbol{I}_n - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$ 是正交矩阵且 $\boldsymbol{P}\boldsymbol{x} = \|\boldsymbol{x}\|_2 \boldsymbol{e}_1$.

**function:** $[v, \beta] = \textbf{house}(\boldsymbol{x})$

$n = \textbf{length}(\boldsymbol{x})$

$\sigma = x(2:n)^{\mathrm{T}} x(2:n)$

$\boldsymbol{v} = \begin{bmatrix} 1 \\ x(2:n) \end{bmatrix}$

**if** $\sigma = 0$

  $\beta = 0$

**else**

  $\mu = \sqrt{x(1)^2 + \sigma}$

  **if** $x(1) \leqslant 0$

   $v(1) = x(1) - \mu$

  **else**

   $v(1) = -\sigma/(x(1) + \mu)$

```
        end
        β = 2v(1)²/(σ + v(1)²)
        v = v/v(1)
    end
```

本算法约需 $3n$ 个 flop, 计算得到的 Householder 矩阵在机器精度内是正交的, 这将在后面讨论. 算法 5.1.1 的实用形式应预先对向量 $\boldsymbol{x}$ 进行加权运算 $(\boldsymbol{x} \leftarrow \boldsymbol{x}/\|\boldsymbol{x}\|)$ 以避免上溢.

### 5.1.4 Householder 矩阵的应用

将 Householder 反射作用于矩阵时, 利用其结构是至关重要的. 如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{P} = \boldsymbol{I} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} \in \mathbb{R}^{m\times m}$, 则

$$\boldsymbol{PA} = (\boldsymbol{I} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}})\boldsymbol{A} = \boldsymbol{A} - \boldsymbol{v}\boldsymbol{w}^{\mathrm{T}},$$

其中 $\boldsymbol{w} = \beta\boldsymbol{A}^{\mathrm{T}}\boldsymbol{v}$. 同样, 如果 $\boldsymbol{P} = \boldsymbol{I} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} \in \mathbb{R}^{n\times n}$, 则

$$\boldsymbol{AP} = \boldsymbol{A}(\boldsymbol{I} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}) = \boldsymbol{A} - \boldsymbol{w}\boldsymbol{v}^{\mathrm{T}},$$

其中 $\boldsymbol{w} = \beta\boldsymbol{A}\boldsymbol{v}$. 这样一个 $m \times n$ Householder 修正由一次矩阵-向量乘法和一次外积运算组成, 它需要 $4mn$ 个 flop. 如果认识不到这一点, 而把 $\boldsymbol{P}$ 当一般矩阵对待就会增加一个数量级的工作量. Householder 修正永远不会出现显式的 Householder 矩阵.

上述两种 Householder 修正中的任何一种实现时都可利用 $v(1) = 1$ 这一事实. 在计算 $\boldsymbol{PA}$ 而 $m$ 很小时和计算 $\boldsymbol{AP}$ 而 $n$ 很小时，这个特性很重要.

作为 Householder 矩阵修正的一个例子, 假设我们想用 $\boldsymbol{B} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}$ 来覆盖 $\boldsymbol{A} \in \mathbb{R}^{m\times n}(m \geqslant n)$, 其中正交矩阵 $\boldsymbol{Q}$ 的选取保证对于满足 $1 \leqslant j \leqslant n$ 的 $j$ 使得 $B(j+1:m, j) = 0$. 此外, 假定 $A(j:m, 1:j-1) = 0$, 且我们打算将 Householder 向量的基本部分存储于 $A(j+1:m, j)$ 中. 下述指令完成这项工作:

$$[v, \beta] = \textbf{house}(A(j:m, j))$$
$$A(j:m, j:n) = (I_{m-j+1} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}})A(j:m, j:n)$$
$$A(j+1:m, j) = v(2:m-j+1)$$

从计算的角度看, 我们用了一个 $m-j+1$ 阶 Householder 矩阵作用于 $\boldsymbol{A}$ 的最下面的 $m-j+1$ 行. 然而在数学上, 我们仍然用了一个 $m \times m$ 的 Householder 矩阵

$$\tilde{\boldsymbol{P}} = \begin{bmatrix} \boldsymbol{I}_{j-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{P} \end{bmatrix} = \boldsymbol{I}_m - \beta\tilde{\boldsymbol{v}}\tilde{\boldsymbol{v}}^{\mathrm{T}}, \quad \tilde{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{v} \end{bmatrix}$$

作用于整个 $\boldsymbol{A}$ 上. 无论怎样, Householder 向量的基本部分可被存储于 $\boldsymbol{A}$ 的化为零的元素之位置上.

### 5.1.5 舍入特性

Householder 矩阵的舍入特性是非常好的.Wilkinson(1965, 第 152~162 页) 指出**house**程序产生的 Householder 向量 $\hat{v}$ 非常接近精确值 $v$. 如果 $\hat{\boldsymbol{P}} = \boldsymbol{I} - 2\hat{\boldsymbol{v}}\hat{\boldsymbol{v}}^{\mathrm{T}}/\hat{\boldsymbol{v}}^{\mathrm{T}}\hat{\boldsymbol{v}}$, 则

$$\|\hat{\boldsymbol{P}} - \boldsymbol{P}\|_2 = O(\mathbf{u}),$$

这意味着 $\hat{\boldsymbol{P}}$ 是在机器精度内正交的. 而且, 用 $\hat{\boldsymbol{P}}$ 的计算修正也非常接近用 $\boldsymbol{P}$ 的精确修正:

$$fl(\hat{\boldsymbol{P}}\boldsymbol{A}) = \boldsymbol{P}(\boldsymbol{A}+\boldsymbol{E}), \quad \|\boldsymbol{E}\|_2 = O(\mathbf{u}\|\boldsymbol{A}\|_2),$$
$$fl(\boldsymbol{A}\hat{\boldsymbol{P}}) = (\boldsymbol{A}+\boldsymbol{E})\boldsymbol{P}, \quad \|\boldsymbol{E}\|_2 = O(\mathbf{u}\|\boldsymbol{A}\|_2).$$

### 5.1.6 因子形式表达

在随后的章节中将给出的许多基于 Householder 变换的分解算法都要计算若干个 Householder 矩阵之乘积:

$$\boldsymbol{Q} = \boldsymbol{Q}_1\boldsymbol{Q}_2\cdots\boldsymbol{Q}_r, \quad \boldsymbol{Q}_j = \boldsymbol{I} - \beta_j v^{(j)}v^{(j)\mathrm{T}}, \tag{5.1.3}$$

其中 $r \leqslant n$, 且每个 $v^{(j)}$ 形如

$$v^{(j)} = (\underbrace{0,0,\cdots,0}_{j-1},1,v_{j+1}^{(j)},\cdots,v_n^{(j)})^{\mathrm{T}}.$$

即便在随后的计算中会涉及 $\boldsymbol{Q}$, 但也没必要将 $\boldsymbol{Q}$ 显式地计算出. 例如, 如果 $\boldsymbol{C} \in \mathbb{R}^{n\times q}$, 我们希望计算 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{C}$, 则只需执行如下循环:

**for** $j = 1:r$
　　$C = Q_jC$
**end**

对 Householder 向量 $v^{(1)}\cdots v^{(r)}$ 和相应的 $\beta_j$ (如果方便) 的存储导致了 $\boldsymbol{Q}$ 的因子形式的表达方式. 为说明因子形式的经济, 假定有数组 $\boldsymbol{A}$, 且 $A(j+1:n,j)$ 中存储第 $j$ 个 Householder 向量的基本部分 $v^{(j)}(j+1:n)$. 以 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{C}$ 覆盖 $\boldsymbol{C} \in \mathbb{R}^{n\times q}$ 的运算可按下述算法执行:

**for** $j = 1:r$

$$v(j:n) = \begin{bmatrix} 1 \\ A(j+1:n,j) \end{bmatrix} \tag{5.1.4}$$

　　$C(j:n,:) = (1 - \beta_j v(j:n)v(j:n)^{\mathrm{T}})C(j:n,:)$
**end**

这需 $2qr(2n-r)$ 个 flop. 如果显式地计算出 $n\times n$ 矩阵 $\boldsymbol{Q}$, 则需 $2n^2q$ 个 flop.

当然在某些应用中必须将 $\boldsymbol{Q}$(或 $\boldsymbol{Q}$ 的一部分) 显式求出. 有两种计算 Householder 乘积矩阵 $\boldsymbol{Q}$ 的算法, 一种是向前累积:

$Q = I_n$
**for** $j = 1:r$
　　$Q = QQ_j$
**end**

另一种是向后累积:

$Q = I_n$
**for** $j = r:-1:1$

$\quad\quad Q = Q_j Q$

$\quad$ **end**

回想到 $\boldsymbol{Q}_j$ 的 $(j-1)\times(j-1)$ 的顺序主子矩阵是单位矩阵. 所以, 在向后累积执行之初, $\boldsymbol{Q}$ 的很大部分是单位矩阵, 它随着迭代的进行逐渐变满. 这个特征可用来减少所需的 flop 数. 相反, 在向前累积法中, 执行完第一步后 $\boldsymbol{Q}$ 就变成满的了. 因此, 向后累积法更为经济, 是应选的方案:

$$
\begin{aligned}
&Q = I_n \\
&\textbf{for } j = r:-1:1 \\
&\quad v(j:n) = \begin{bmatrix} 1 \\ A(j+1:n,j) \end{bmatrix} \\
&\quad Q(j:n,j:n) = (I - \beta_j v(j:n)v(j:n)^{\mathrm{T}})Q(j:n,j:n) \\
&\textbf{end}
\end{aligned} \tag{5.1.5}
$$

这需约 $4(n^2r - nr^2 + r^3/3)$ 个 flop.

### 5.1.7 分块形式

假设 $\boldsymbol{Q} = \boldsymbol{Q}_1 \cdots \boldsymbol{Q}_r$ 是几个 $n\times n$ Householder 矩阵的乘积, 如 (5.1.3). 由于 $\boldsymbol{Q}_j$ 是单位矩阵的秩 1 校正, 由 Householder 向量的结构, $\boldsymbol{Q}$ 是单位矩阵的秩 $r$ 校正, 可以写成

$$\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}}, \tag{5.1.6}$$

其中 $\boldsymbol{W}$ 和 $\boldsymbol{Y}$ 是 $n\times r$ 矩阵. 计算分块形式(5.1.6) 的关键在于下面引理.

**引理 5.1.1** 假定 $\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}}$ 是 $n\times n$ 的正交矩阵, 其中 $\boldsymbol{W}, \boldsymbol{Y} \in \mathbb{R}^{n\times j}$. 如果 $\boldsymbol{P} = \boldsymbol{I} - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$, 其中 $\boldsymbol{v}\in\mathbb{R}^n, \boldsymbol{z} = -\beta\boldsymbol{Q}\boldsymbol{v}$, 则

$$\boldsymbol{Q}_+ = \boldsymbol{Q}\boldsymbol{P} = \boldsymbol{I} + \boldsymbol{W}_+\boldsymbol{Y}_+^{\mathrm{T}}$$

其中 $\boldsymbol{W}_+ = [\boldsymbol{W}\ z]$ 和 $\boldsymbol{Y}_+ = [\boldsymbol{Y}\ v]$ 均为 $n\times(j+1)$ 矩阵.

**证明**

$$
\begin{aligned}
\boldsymbol{QP} &= (\boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}})(I - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}) = \boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}} - \beta\boldsymbol{Q}\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} \\
&= \boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}} + \boldsymbol{z}\boldsymbol{v}^{\mathrm{T}} = \boldsymbol{I} + [\boldsymbol{W}\ \boldsymbol{z}][\boldsymbol{Y}\ \boldsymbol{v}]^{\mathrm{T}}.
\end{aligned}
$$

□

反复应用这个引理, 可由 (5.1.3) 中的 $\boldsymbol{Q}$ 的因子形式得到分块形式如下:

**算法 5.1.2** 假定 $\boldsymbol{Q} = \boldsymbol{Q}_1 \cdots \boldsymbol{Q}_r$ 是如 (5.1.3) 所示的一系列 $n\times n$ Householder 矩阵的乘积, 本算法计算满足 $\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}}$ 的矩阵 $\boldsymbol{W}, \boldsymbol{Y} \in \mathbb{R}^{n\times r}$.

$$
\begin{aligned}
&Y = v^{(1)} \\
&W = -\beta_1 v^{(1)} \\
&\textbf{for } j = 2:r \\
&\quad z = -\beta_j(I + WY^{\mathrm{T}})v^{(j)} \\
&\quad W = [W\ z] \\
&\quad Y = [Y\ v^{(j)}] \\
&\textbf{end}
\end{aligned}
$$

如果利用了 $v^{(j)}$ 中的零元素, 则本算法需要 $2r^2n-2r^3/3$ 个 flop. 注意到 $\boldsymbol{Y}$ 仅仅是由 Householder 向量组成的矩阵, 因此, 它是单位下三角形矩阵. 很显然, 生成 $\boldsymbol{WY}$ 表示形式 (5.1.6) 的中心任务是计算 $\boldsymbol{W}$ 矩阵.

当 $\boldsymbol{Q}$ 与另一个矩阵进行运算时, Householder 矩阵的乘积的分块形式就显得很有吸引力. 假定 $\boldsymbol{C}\in\mathbb{R}^{n\times q}$, 则运算

$$\boldsymbol{C}\leftarrow\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{C}=(\boldsymbol{I}+\boldsymbol{WY}^{\mathrm{T}})^{\mathrm{T}}\boldsymbol{C}=\boldsymbol{C}+\boldsymbol{Y}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{C})$$

含有大量的 3 级运算. 另一方面, 如果 $\boldsymbol{Q}$ 是因子形式, $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{C}$ 只是包含大量的矩阵与向量相乘和外积修正等 2 级运算. 当然, 关于此, 随着 $\boldsymbol{C}$ 列数的减少, 2 级运算和 3 级运算之间的差别也逐渐消失.

我们想说明的是, 从几何上说, $\boldsymbol{WY}$ 表示不是一般化的 Householder 变换. 真正的分块表示的形式为 $\boldsymbol{Q}=\boldsymbol{I}-2\boldsymbol{VV}^{\mathrm{T}}$, 其中 $\boldsymbol{V}\in\mathbb{R}^{n\times r}$ 满足 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{V}=\boldsymbol{I}_r$, 参阅 Schreiber and Parlett(1987) 及 Schreiber and Van Loan(1989).

**例 5.1.3** 如果 $n=4, r=2$, 且 $[1,0.6,0,0.8]^{\mathrm{T}}$, $[0,1,0.8,0.6]^{\mathrm{T}}$ 分别是与 $Q_1,Q_2$ 相关的 Householder 向量, 则

$$\boldsymbol{Q}_1\boldsymbol{Q}_2=\boldsymbol{I}_4+\boldsymbol{WY}^{\mathrm{T}}\equiv\boldsymbol{I}_4+\begin{bmatrix}-1 & 1.080\\ -0.6 & -0.352\\ 0 & -0.800\\ -0.8 & 0.264\end{bmatrix}\begin{bmatrix}1 & 0.6 & 0 & -0.8\\ 0 & 1 & 0.8 & 0.6\end{bmatrix}.$$

### 5.1.8 Givens 旋转

Householder 反射对于大量引进零元是非常有用的, 例如, 消去一个向量中除第一个分量外的所有分量. 然而, 在许多计算中, 必须有选择地消去一些元素. Givens旋转就是解决这个问题的工具. 下面这些矩阵是单位矩阵的秩 2 校正:

$$G(i,k,\theta)=\begin{bmatrix}1 & \cdots & 0 & \cdots & 0 & \cdots & 0\\ \vdots & \ddots & \vdots & & \vdots & & \vdots\\ 0 & \cdots & c & \cdots & s & \cdots & 0\\ \vdots & & \vdots & \ddots & \vdots & & \vdots\\ 0 & \cdots & -s & \cdots & c & \cdots & 0\\ \vdots & & \vdots & & \vdots & \ddots & \vdots\\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 0\end{bmatrix}\begin{matrix}\\ \\ i\\ \\ k\\ \\ \\ \end{matrix}\qquad(5.1.7)$$

$$\qquad\qquad\qquad\qquad i\qquad\qquad k$$

其中对于某个 $\theta$, $c=\cos(\theta), s=\sin(\theta)$, 显然 Givens 旋转是正交变换.

用 $G(i,k,\theta)^{\mathrm{T}}$ 进行左乘产生一个在 $(i,k)$ 坐标平面的 $\theta$ 弧度的逆时针旋转. 实际上, 如果 $\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{y}=G(i,k,\theta)^{\mathrm{T}}\boldsymbol{x}$, 则

$$y_j=\begin{cases}cx_i-sx_k, & j=i,\\ sx_i+cx_k, & j=k,\\ x_j, & j\neq i,k.\end{cases}$$

从这组公式很清楚知道, 我们可以通过令

$$c=\frac{x_i}{\sqrt{x_i^2+x_k^2}},\quad s=\frac{-x_k}{\sqrt{x_i^2+x_k^2}},\tag{5.1.8}$$

而使 $y_k$ 为 0. 因此, 使用 Givens 旋转很容易就可将一向量的某个指定分量化为 0. 实际上还有比 (5.1.8) 更好的方法来计算 $c$ 和 $s$. 例如, 下述算法能防止溢出发生.

**算法 5.1.3** 给定标量 $a$ 和 $b$, 本函数计算 $c=\cos(\theta)$ 和 $s=\sin(\theta)$, 使得

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

**function:** $[c,s]=$ **givens**$(a,b)$
    **if** $b=0$
        $c=1; s=0;$
    **else**
        **if** $|b|>|a|$
            $\tau=-a/b; s=1/\sqrt{1+\tau^2}; c=s\tau$
        **else**
            $\tau=-b/a; c=1/\sqrt{1+\tau^2}; s=c\tau$
        **end**
    **end**

本算法需要 5 个 flop 和一次求平方根运算. 注意其中并没有计算 $\theta$, 因此不涉及反三角函数.

**例 5.1.4** 如果 $\boldsymbol{x}=[1,2,3,4]^{\mathrm{T}}, \cos(\theta)=1/\sqrt{5}, \sin(\theta)=-2/\sqrt{5}$, 则 $G(2,4,\theta)\boldsymbol{x}=(1,\sqrt{20},3,0)^{\mathrm{T}}$.

### 5.1.9 Givens 旋转的应用

当应用 Givens 旋转矩阵做乘法运算时, 关键是要利用它的简单结构. 假定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}, c=\cos(\theta), s=\sin(\theta)$. 如果 $G(i,k,\theta)\in\mathbb{R}^{m\times m}$, 则修正 $\boldsymbol{A}\leftarrow G(i,k,\theta)^{\mathrm{T}}\boldsymbol{A}$ 只影响 $\boldsymbol{A}$ 的两行

$$A([i,k],:)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} A([i,k],:),$$

且只需 $6n$ 个 flop:

**for** $j=1:n$
    $\tau_1=A(i,j)$
    $\tau_2=A(k,j)$
    $A(i,j)=c\tau_1-s\tau_2$
    $A(k,j)=s\tau_1+c\tau_2$
**end**

同样, 如果 $G(i,k,\theta)\in\mathbb{R}^{n\times n}$, 则修正 $\boldsymbol{A}\leftarrow \boldsymbol{A}G(i,k,\theta)$ 只影响 $\boldsymbol{A}$ 的两列

$$A(:,[i,k]) = A(:,[i,k]) \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

且只需 $6m$ 个 flop:

**for** $j = 1:m$

  $\tau_1 = A(j,i)$

  $\tau_2 = A(j,k)$

  $A(j,i) = c\tau_1 - s\tau_2$

  $A(j,k) = s\tau_1 + c\tau_2$

**end**

### 5.1.10 舍入特性

Givens 旋转的数值性质与 Householder 反射一样良好. 确切地说, 可以证明 givens 程序的计算解 $\hat{c}$ 和 $\hat{s}$ 满足

$$\hat{c} = c(1+\varepsilon_c), \quad \varepsilon_c = O(\mathbf{u}),$$
$$\hat{s} = s(1+\varepsilon_s), \quad \varepsilon_s = O(\mathbf{u}).$$

如果接着用 $\hat{c}$ 和 $\hat{s}$ 进行 Givens 修正, 则计算所得的修正是一个近似矩阵精确修正:

$$fl[\hat{G}(i,k,\theta)^{\mathrm{T}}\boldsymbol{A}] = G(i,k,\theta)^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E}), \quad \|\boldsymbol{E}\|_2 \approx \mathbf{u}\|\boldsymbol{A}\|_2,$$
$$fl[\boldsymbol{A}\hat{G}(i,k,\theta)] = (\boldsymbol{A}+\boldsymbol{E})G(i,k,\theta), \quad \|\boldsymbol{E}\|_2 \approx \mathbf{u}\|\boldsymbol{A}\|_2.$$

Givens 旋转详细的误差分析可在 Wilkinson(1965, 第 131—139 页) 中查到.

### 5.1.11 Givens 旋转的表示积

假定 $\boldsymbol{Q} = \boldsymbol{G}_1 \cdots \boldsymbol{G}_t$ 是若干个 Givens 旋转的乘积. 正如关于 Householder 反射的结论一样, 将正交矩阵 $\boldsymbol{Q}$ 保持为因子形式要比显式地求出其乘积更为经济. 应用 Stewart(1976) 提出的技术, 可用一简洁的方法做到这一点. 其思想是将每一个旋转变换对应于一个浮点数 $\rho$. 具体地说, 如果

$$\boldsymbol{Z} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}, \quad c^2 + s^2 = 1,$$

则定义标量 $\rho$ 如下:

**if** $c = 0$

  $\rho = 1$

**elseif** $|s| < |c|$

  $\rho = \mathrm{sign}(c)s/2$ (5.1.9)

**else**

  $\rho = 2\mathrm{sign}(s)/c$

**end**

实质上, 相当于当正弦较小时存储 $s/2$, 余弦较小时存储 $2/c$. 按这种对应关系, 可以重新构造 $\pm\boldsymbol{Z}$ 如下:

$$
\begin{array}{l}
\textbf{if } \rho = 1 \\
\quad c = 0; s = 1 \\
\textbf{elseif } |\rho| < 1 \\
\quad s = 2\rho; c = \sqrt{1-s^2} \\
\textbf{else} \\
\quad c = 2\rho; s = \sqrt{1-c^2} \\
\textbf{end}
\end{array}
\tag{5.1.10}
$$

这样也许会产生 $-\boldsymbol{Z}$, 不过这无关紧要, 因为如果 $\boldsymbol{Z}$ 能使矩阵的某个元素变为 0, 则 $-\boldsymbol{Z}$ 也能做到. 把 $c$ 和 $s$ 中的较小的数存储起来的最根本理由是, 如果 $x$ 接近 1, 公式 $\sqrt{1-x^2}$ 的精度就低得可怜. 在 Stewart(1976) 中可以找到更多的细节. 当然, 为重构 $G(i,k,\theta)$, 除需相关的 $\rho$ 外, 还需 $i$ 和 $k$ 的值. 我们将在 5.2.3 节中讨论, 这并没有什么困难.

### 5.1.12 误差传播

我们简要讨论当算法涉及一系列 Householder 修正或 Givens 修正时舍入误差的传播. 精确地说, 假定给出 $\boldsymbol{A} = \boldsymbol{A}_0 \in \mathbb{R}^{m\times n}$, 并且矩阵 $\boldsymbol{A}_1 \cdots \boldsymbol{A}_p = \boldsymbol{B}$ 是利用

$$\boldsymbol{A}_k = fl(\hat{\boldsymbol{Q}}_k \boldsymbol{A}_{k-1} \hat{\boldsymbol{Z}}_k), \quad k = 1:p$$

产生的. 假定 $\hat{\boldsymbol{Q}}_k$ 和 $\hat{\boldsymbol{Z}}_k$ 的产生及应用都使用上述的 Householder 算法和 Givens 算法. 设 $\boldsymbol{Q}_k$ 和 $\boldsymbol{Z}_k$ 是没有舍入误差的情况下产生的正交矩阵, 能够证明

$$\boldsymbol{B} = (\boldsymbol{Q}_p \cdots \boldsymbol{Q}_1)(\boldsymbol{A}+\boldsymbol{E})(\boldsymbol{Z}_1 \cdots \boldsymbol{Z}_p), \tag{5.1.11}$$

其中 $\|\boldsymbol{E}\|_2 \leqslant c\mathbf{u}\|\boldsymbol{A}\|_2$, $c$ 是一个温和依赖于 $n, m$ 和 $p$ 的常数. 简言之, $\boldsymbol{B}$ 是某个近似于 $\boldsymbol{A}$ 的矩阵之精确的正交修正.

### 5.1.13 快速 Givens 变换

由于 Givens 旋转能够有选择地引进零元, 这使得它在一些有特殊结构的问题中成为一个重要的消零工具. 这导致了"快速 Givens"方法的发展. 快速 Givens 思想就是当 $\boldsymbol{Q}$ 是一系列 Givens 旋转的乘积时巧妙地表达它. 具体地说, 就是用一个矩阵对 $(\boldsymbol{M}, \boldsymbol{D})$ 来表示 $\boldsymbol{Q}$, 其中 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} = \boldsymbol{D} = \mathrm{diag}(d_i)$, 且每个 $d_i$ 都大于 0. 矩阵 $\boldsymbol{Q}, \boldsymbol{M}$ 和 $\boldsymbol{D}$ 通过公式

$$\boldsymbol{Q} = \boldsymbol{M}\boldsymbol{D}^{-1/2} = \boldsymbol{M}\mathrm{diag}(1/\sqrt{d_i})$$

联系起来. 请注意 $(\boldsymbol{M}\boldsymbol{D}^{-1/2})^{\mathrm{T}}(\boldsymbol{M}\boldsymbol{D}^{-1/2}) = \boldsymbol{D}^{-1/2}\boldsymbol{D}\boldsymbol{D}^{-1/2} = \boldsymbol{I}$, 因此矩阵 $\boldsymbol{M}\boldsymbol{D}^{-1/2}$ 是正交的. 而且如果 $\boldsymbol{F}$ 是 $n \times n$ 矩阵且 $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{F} = \boldsymbol{D}_{new}$ 为对角矩阵, 则 $\boldsymbol{M}_{new}^{\mathrm{T}}\boldsymbol{M}_{new} = \boldsymbol{D}_{new}$, 其中 $\boldsymbol{M}_{new} = \boldsymbol{M}\boldsymbol{F}$. 因此可以通过对快速 Givens 的表示形式 $(\boldsymbol{M}, \boldsymbol{D})$ 做修正来得到 $(\boldsymbol{M}_{new}, \boldsymbol{D}_{new})$. 为使这个思想具有实际应用的意义, 我们必须给出如何在保证 $\boldsymbol{D}$ 为对角矩阵的限制条件下使 $\boldsymbol{F}$ 具有消零的能力.

在 2×2 的层次上能很好地给出详细解释. 令 $\boldsymbol{x} = (x_1, x_2)^{\mathrm{T}}$, 且给定 $\boldsymbol{D} = \mathrm{diag}(d_1, d_2)$, 假定 $d_1$ 和 $d_2$ 大于零. 定义

$$M_1 = \begin{bmatrix} \beta_1 & 1 \\ 1 & \alpha_1 \end{bmatrix}, \tag{5.1.12}$$

且观察到

$$M_1^{\mathrm{T}} x = \begin{bmatrix} \beta_1 x_1 + x_2 \\ x_1 + \alpha_1 x_2 \end{bmatrix},$$

$$M_1^{\mathrm{T}} D M_1 = \begin{bmatrix} d_2 + \beta_1^2 d_1 & d_1\beta_1 + d_2\alpha_1 \\ d_1\beta_1 + d_2\alpha_1 & d_1 + \alpha_1^2 d_2 \end{bmatrix} \equiv D_1.$$

如果 $x_2 \neq 0, \alpha_1 = -x_1/x_2$, 且 $\beta_1 = -\alpha_1 d_2/d_1$, 则

$$M_1^{\mathrm{T}} x = \begin{bmatrix} x_2(1+\gamma_1) \\ 0 \end{bmatrix},$$

$$M_1^{\mathrm{T}} D M_1 = \begin{bmatrix} d_2(1+\gamma_1) & 0 \\ 0 & d_1(1+\gamma_1) \end{bmatrix},$$

其中 $\gamma_1 = -\alpha_1\beta_1 = (d_2/d_1)(x_1/x_2)^2$.

类似地, 如我们假定 $x_1 \neq 0$, 且定义 $M_2$ 如下:

$$M_2 = \begin{bmatrix} 1 & \alpha_2 \\ \beta_2 & 1 \end{bmatrix}, \tag{5.1.13}$$

其中 $\alpha_2 = -x_2/x_1$, 且 $\beta_2 = -(d_1/d_2)\alpha_2$, 则

$$M_2^{\mathrm{T}} x = \begin{bmatrix} x_1(1+\gamma_2) \\ 0 \end{bmatrix},$$

$$M_2^{\mathrm{T}} D M_2 = \begin{bmatrix} d_1(1+\gamma_2) & 0 \\ 0 & d_2(1+\gamma_2) \end{bmatrix} \equiv D_2,$$

其中 $\gamma_2 = -\alpha_2\beta_2 = (d_1/d_2)(x_2/x_1)^2$.

容易证明, 无论 $i=1$ 或 $i=2$, 矩阵 $J = D^{1/2} M_i D_i^{-1/2}$ 都是正交矩阵, 且它能使 $J^{\mathrm{T}}(D^{-1/2}x)$ 的第二个分量为 0. ($J$ 也许实际上是一个反射变换, 因此采用流行的名称“快速 Givens 变换”算是对了一半.)

注意到 $\gamma_i$ 满足 $\gamma_1\gamma_2 = 1$. 因此, 我们选择上述的 $M_i$ 使“增长因子”$(1+\gamma_i)$ 以 2 为界. 形如

$$M_1 = \begin{bmatrix} \beta_1 & 1 \\ 1 & \alpha_1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 & \alpha_2 \\ \beta_2 & 1 \end{bmatrix}$$

并满足 $-1 \leqslant \alpha_i\beta_i \leqslant 0$ 的矩阵是 $2\times 2$ 的快速Givens变换. 注意, 用快速 Givens 变换左乘, 所需乘法次数仅为“普通”的 Givens 变换的一半. 而且, 消零过程并不需计算平方根.

在 $n \times n$ 的情形下，一切都和普通的 Givens 旋转一样"放大"．"1 型"变换的形式为

$$F(i,k,\alpha,\beta)=\begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \beta & \cdots & 1 & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & 1 & \cdots & \alpha & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \begin{matrix} \\ \\ i \\ \\ k \\ \\ \\ \end{matrix} \tag{5.1.14}$$
$$\qquad\qquad\qquad i \qquad\quad k$$

而"2 型"变换的结构为

$$F(i,k,\alpha,\beta)=\begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & 1 & \cdots & \alpha & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \beta & \cdots & 1 & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \begin{matrix} \\ \\ i \\ \\ k \\ \\ \\ \end{matrix} \tag{5.1.15}$$
$$\qquad\qquad\qquad i \qquad\quad k$$

综合这些可得下列算法.

**算法 5.1.4**　给定 $\boldsymbol{x} \in \mathbb{R}^2$ 和正的 $\boldsymbol{d} \in \mathbb{R}^2$，本算法计算一个 $2 \times 2$ 快速 Givens 变换 $\boldsymbol{M}$，使得 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{x}$ 的第二个分量为零且 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{M} = \boldsymbol{D}_1$ 是对角矩阵，其中 $\boldsymbol{D} = \mathrm{diag}(d_1, d_2)$．如果 type = 1，则 $\boldsymbol{M}$ 的形式为 (5.1.12)；如果 type = 2，则 $\boldsymbol{M}$ 的形式为 (5.1.13)．$\boldsymbol{D}_1$ 对角元素覆盖 $\boldsymbol{d}$.

```
function: [α, β, type] = fast.givens(x, d)
    if x(2) ≠ 0
        α = −x(1)/x(2); β = −αd(2)/d(1); γ = −αβ
        if γ ≤ 1
            type=1
            τ = d(1); d(1) = (1 + γ)d(2); d(2) = (1 + γ)τ
        else
            type=2
            α = 1/α; β = 1/β; γ = 1/γ
            d(1) = (1 + γ)d(1); d(2) = (1 + γ)d(2)
        end
    else
        type=2
```

$\alpha = 0; \beta = 0$

**end**

快速 Givens 变换的应用类似于普通的 Givens 变换. 即使选择了适当的变换类型, 增长因子 $1+\gamma$ 仍可能达到 2. 因此, 经过 $s$ 次修正后 $\boldsymbol{D}$ 和 $\boldsymbol{M}$ 的元素可能已增加了 $2^s$. 这意味着在进行快速 Givens 变换过程中必须监控对角矩阵 $\boldsymbol{D}$ 以防溢出. 参阅 Anda and Park(1994), 看如何有效做到这一点.

尽管如此, 因为 $\boldsymbol{M}\boldsymbol{D}^{-1/2}$ 总是正交的, 故 $\boldsymbol{M}$ 和 $\boldsymbol{D}$ 的元素增长总是受控的. 快速 Givens 方法的舍入性质是我们对 Givens 矩阵技术所期望的. 例如, 如果我们计算 $\hat{\boldsymbol{Q}} = fl(\hat{\boldsymbol{M}}\hat{\boldsymbol{D}}^{-1/2})$, 其中 $\hat{\boldsymbol{M}}$ 和 $\hat{\boldsymbol{D}}$ 是 $\boldsymbol{M}$ 和 $\boldsymbol{D}$ 的计算解, 则 $\boldsymbol{Q}$ 是达到工作精度 $\|\hat{\boldsymbol{Q}}^{\mathrm{T}}\hat{\boldsymbol{Q}} - \boldsymbol{I}\|_2 \approx \mu$ 的正交矩阵.

## 习 题

**5.1.1** 对 $\boldsymbol{x} = [1, 7, 2, 3, -1]^{\mathrm{T}}$ 执行 house.

**5.1.2** 令 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是 $\mathbb{R}^n$ 上的非零向量, 给出一个算法求 Householder 矩阵 $\boldsymbol{P}$, 使得 $\boldsymbol{P}\boldsymbol{x}$ 是 $\boldsymbol{y}$ 的数乘.

**5.1.3** 假设 $\boldsymbol{x} \in \mathbb{C}^n, x_1 = |x_1|\mathrm{e}^{\mathrm{i}\theta}$, 其中 $\theta \in \mathbb{R}$. 假定 $\boldsymbol{x} \neq \boldsymbol{0}$, 定义 $\boldsymbol{u} = \boldsymbol{x} + \mathrm{e}^{\mathrm{i}\theta}\|x_2\|\boldsymbol{e}_1$, 证明 $\boldsymbol{P} = \boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^H/\boldsymbol{u}^H\boldsymbol{u}$ 是酉矩阵且有 $\boldsymbol{P}\boldsymbol{x} = -\mathrm{e}^{\mathrm{i}\theta}\|\boldsymbol{x}\|_2\boldsymbol{e}_1$.

**5.1.4** 运用 Householder 矩阵证明 $\det(\boldsymbol{I} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}) = 1 + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$, 其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是 $n$ 维向量.

**5.1.5** 假定 $\boldsymbol{x} \in \mathbb{C}^2$, 试给出一个算法产生一个形如

$$\boldsymbol{Q} = \begin{bmatrix} c & \bar{s} \\ -s & c \end{bmatrix}, \quad c \in \mathbb{R}, \quad c^2 + |s|^2 = 1$$

的酉矩阵, 使 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{x}$ 的第二个分量为 0

**5.1.6** 假设 $\boldsymbol{x}, \boldsymbol{y}$ 是 $\mathbb{R}^n$ 上的单位向量, 试给出一个算法, 要求该算法使用 Givens 变换来计算出一个正交矩阵 $\boldsymbol{Q}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{y}$.

**5.1.7** 确定 $c = \cos(\theta)$ 和 $s = \sin(\theta)$ 使得

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 5 \\ 12 \end{bmatrix} = \begin{bmatrix} 13 \\ 0 \end{bmatrix}.$$

**5.1.8** 假定 $\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{Y}\boldsymbol{T}\boldsymbol{Y}^{\mathrm{T}}$ 是正交矩阵且 $\boldsymbol{Y} \in \mathbb{R}^{n\times j}$ 和 $\boldsymbol{T} \in \mathbb{R}^{j\times j}$ 是上三角形矩阵. 证明: 如果 $\boldsymbol{Q}_+ = \boldsymbol{Q}\boldsymbol{P}$, 其中 $\boldsymbol{P} = \boldsymbol{I} - 2\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}/\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}$ 是 Householder 矩阵, 则 $\boldsymbol{Q}_+$ 可被表示为 $\boldsymbol{Q}_+ = \boldsymbol{I} + \boldsymbol{Y}_+\boldsymbol{T}_+\boldsymbol{Y}_+^{\mathrm{T}}$, 其中 $\boldsymbol{Y}_+ \in \mathbb{R}^{n\times(j+1)}$ 和 $\boldsymbol{T}_+ \in \mathbb{R}^{(j+1)\times(j+1)}$ 是上三角形矩阵.

**5.1.9** 假设第 $j$ 个 Householder 向量的基本部分 $v^{(j)}(j+1:n)$ 存储于 $A(j+1:n, j)$ 中, 给出算法 5.1.2 的一个具体实现. 由于 $\boldsymbol{A}$ 中已经有效地表示了 $\boldsymbol{Y}$, 所以你只需对 $\boldsymbol{W}$ 矩阵做好安排.

**5.1.10** 证明: 如果 $\boldsymbol{S}$ 是反对称矩阵 $(\boldsymbol{S}^{\mathrm{T}} = -\boldsymbol{S})$, 则 $\boldsymbol{Q} = (\boldsymbol{I} + \boldsymbol{S})(\boldsymbol{I} - \boldsymbol{S})^{-1}$ 是正交矩阵. ($\boldsymbol{Q}$ 被称为是 $\boldsymbol{S}$ 的 Cayley 变换). 构造一个秩为 2 的矩阵 $S$, 使得 $\boldsymbol{Q}\boldsymbol{x}$ 除第一个元素外均为零, $\boldsymbol{x}$ 是向量.

**5.1.11** 假设 $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ 满足 $\|\boldsymbol{P}^{\mathrm{T}}\boldsymbol{P} - \boldsymbol{I}_n\|_2 = \varepsilon < 1$, 证明 $\boldsymbol{P}$ 的所有奇异值均在区间 $[1-\varepsilon, 1+\varepsilon]$ 中, 且 $\|\boldsymbol{P} - \boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}\|_2 \leqslant \varepsilon$, 其中 $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$ 是 $\boldsymbol{P}$ 的奇异值分解.

**5.1.12** 假定 $A \in \mathbb{R}^{2\times 2}$, 在什么条件下 $A$ 的最近旋转比最近反射更近?

## 本节注释与参考文献

Householder 矩阵以 A. S. Householder 命名, 他使得 Householder 矩阵在数值分析中得到广泛应用. 然而, 此类矩阵的某些性质早就被人们所知了, 参见:

H. W. Turnbull and A. C. Aitken(1961). *An Introduction to the Theory of Canonical Matrices*, Dover Publications, New York, pp. 102–105.

有关 Householder 变换的其他文献包括:

A. R. Gourlay (1970). "Generalization of Elementary Hermitian Matrices," *Comp. J.* 13, 411–412.

B. N. Parlett(1971). "Analysis of Algorithms for Reflections in Bisectors," *SIAM Review* 13, 197–208.

N. K. Tsao(1975). "A Note on Implementing the Householder Transformations," *SIAM J. Num. Anal.* 12, 53–58.

B. Danloy(1976). "On the Choice of Signs for Householder Matrices," *J. Comp. Appl. Math.* 2, 67–69.

J. J. M. Cuppen (1984). "On Updating Triangular Products of Householder Matrices," *Numer. Math.* 45, 403–410.

L. Kaufman(1987). "The Generalized Householder Transformation and Sparse Matrices," *Lin. Alg. and Its Applic.* 90, 221–234.

更详尽的 Householder 变换的误差分析见 Lawson and Hanson(1974, 83–89). 分块 Householder 表示和相关计算的主要参考包括:

C. H. Bischof and C. Van Loan(1987). "The WY Representation for Products of Householder Matrices," *SIAM J. Sci. and Stat. Comp.* 8, s2–s13.

R. Schreiber and B. N. Parlett(1987). "Block Reflectors: Theory and Computation," *SIAM J. Numer. Anal.* 25, 189–205.

B. N. Parlett and R. Schreiber(1988). "Block Reflectors: Theory and Computation," *SIAM J. Num. Anal.* 25, 189–205.

R. S. Schreiber and C. Van Loan(1989). "A Storage-Efficient WY Representation for Products of Householder Transformations," *SIAM J. Sci. and Stat. Comp.* 10, 52–57.

C. Puglisi(1992). "Modification of the Householder Method Based on the Compact WY Representation," *SIAM J. Sci, and Stat. Comp.* 13, 723–726.

X. Sun and C. H. Bischof(1995). "A Basis-Kernel Representation of Orthogonal Matrices," *SIAM J. Matrix Anal. Appl.* 16, 1184–1196.

以 W. Givens 命名的 Givens 旋转又被称为 Jacobi 旋转. Jacobi 在 1846 年以这些变换为基础设计了对称特征值算法, 参阅 8.4 节. Given 旋转的存储方法的详细讨论参阅:

G. W. Stewart(1976). "The Economical Storage of Plane Rotations", *Numer. Math.* 25, 137–138.

快速 Givens 变换又被称作"免求平方根"运算,(回想一下,Given 变换的形成都需经过求平方根运算),有好几种方法来安排快速 Givens 运算. 见:

M. Gentleman(1973). "Least Squares Computations by Givens Transformations without Square Roots," *J. Inst. Math. Appl.* 12, 329–336.

C. F. Van Loan(1973). "Generalized Singular Values With Algorithms and Applications," Ph. D. thesis, University of Michigan, Ann Arbor.

S. Hammarling (1974). "A Note on Modifications to the Givens Plane Rotation," *J. Inst. Math. Appl.* 13, 215-218.

J. H. Wilkinson(1977). "Some Recent Advances in Numerical Linear Algebra," in *The State of the Art in Numerical Analysis*, ed. D. A. H. Jacobs, Academic Press, New York, pp. 1–53.

A. A. Anda and H. Park(1994). "Fast Plane Rotations with Dynamic Scaling," *SIAM J. Matrix Anal. Appl.* 15, 162–174.

## 5.2 QR 分解

我们现在叙述如何应用 Householder 变换和 Givens 变换来计算各种分解, 首先是 QR 分解. 一个 $m \times n$ 矩阵 $A$ 的 QR 分解为

$$A = QR,$$

其中 $Q \in \mathbb{R}^{m\times m}$ 是正交矩阵, $R \in \mathbb{R}^{m\times n}$ 是上三角形矩阵. 本节中我们假定 $m \geqslant n$. 我们将会看到如果 $A$ 是列满秩的, 则 $Q$ 的前 $n$ 列形成 $\mathrm{ran}(A)$ 的一组正交基. 因此, 计算 QR 分解也是求解一组向量之正交基的一种方法. 此计算可按几种方法进行. 我们给出的方法包括基于 Householder 变换、分块 Householder 变换、Givens 变换和快速 Givens 变换. 同时还讨论了 Gram-Schmidt 正交化方法和一种在数值上更稳定的所谓修正 Gram-Schmidt 方法.

### 5.2.1 Householder QR 分解

我们首先讨论利用 Householder 变换的 QR 分解方法. 算法的要旨可通过一个小例子来展示. 考虑 $m = 6, n = 5$, 假定 Householder 矩阵 $H_1$ 和 $H_2$ 已算出, 它们使得

$$H_2H_1A = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \otimes & \times & \times \\ 0 & 0 & \otimes & \times & \times \\ 0 & 0 & \otimes & \times & \times \\ 0 & 0 & \otimes & \times & \times \end{bmatrix}.$$

将注意力集中在这些被标记的元素上, 我们要给出一个 Householder 矩阵 $\tilde{H}_3 \in \mathbb{R}^{4\times 4}$ 使得

$$\tilde{\boldsymbol{H}}_3 \begin{bmatrix} \otimes \\ \otimes \\ \otimes \\ \otimes \end{bmatrix} = \begin{bmatrix} \times \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

如果 $\boldsymbol{H}_3 = \text{diag}(\boldsymbol{I}_2, \tilde{\boldsymbol{H}}_3)$, 则

$$\boldsymbol{H}_3\boldsymbol{H}_2\boldsymbol{H}_1\boldsymbol{A} = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

照此执行 $n$ 步就得到上三角形矩阵 $\boldsymbol{H}_n\boldsymbol{H}_{n-1}\cdots\boldsymbol{H}_1\boldsymbol{A} = \boldsymbol{R}$, 令 $\boldsymbol{Q} = \boldsymbol{H}_1\cdots\boldsymbol{H}_n$ 我们得到 $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$.

**算法 5.2.1 (Householder QR)**　给定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, m \geqslant n$, 本算法计算 Householder 矩阵 $\boldsymbol{H}_1\cdots\boldsymbol{H}_n$ 满足: 如果 $\boldsymbol{Q} = \boldsymbol{H}_1\cdots\boldsymbol{H}_n$, 则 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{R}$ 是上三角形矩阵. $\boldsymbol{A}$ 的上三角部分被 $\boldsymbol{R}$ 的上三角部分覆盖, 第 $j$ 个 Householder 向量的 $j+1$: $m$ 分量存储于 $A(j+1:m, j), j < m$.

```
for j = 1 : n
    [v, β] = house(A(j : m, j))
    A(j : m, j : n) = (I_{m-j+1} − βvv^T)A(j : m, j : n)
    if j < m
        A(j + 1 : m, j) = v(2 : m − j + 1)
    end
end
```

本算法需要 $2n^2(m-n/3)$ 个 flop.

为说明 $\boldsymbol{A}$ 是如何被覆盖的, 设

$$v^{(j)} = [\underbrace{0, 0, \cdots 0}_{j-1}, 1,\ v_{j+1}^{(j)}, \cdots, v_m^{(j)}]^{\mathrm{T}}$$

是第 $j$ 个 Householder 向量, 则算法执行完毕时

$$\boldsymbol{A} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ v_2^{(1)} & r_{22} & r_{23} & r_{24} & r_{25} \\ v_3^{(1)} & v_3^{(2)} & r_{33} & r_{34} & r_{35} \\ v_4^{(1)} & v_4^{(2)} & v_4^{(3)} & r_{44} & r_{45} \\ v_5^{(1)} & v_5^{(2)} & v_5^{(3)} & v_5^{(4)} & r_{55} \\ v_6^{(1)} & v_6^{(2)} & v_6^{(3)} & v_6^{(4)} & v_6^{(5)} \end{bmatrix}$$

如果需求出矩阵 $\boldsymbol{Q}=\boldsymbol{H}_1\cdots\boldsymbol{H}_n$, 则可由 (5.1.5) 而累积得到. 累积需 $4(m^2n-mn^2+n^3/3)$ 个 flop.

所计算的上三角形矩阵 $\boldsymbol{R}$ 在满足 $\boldsymbol{Z}^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E})=\hat{\boldsymbol{R}}$ 的意义下是一个附近的 $\boldsymbol{A}$ 之精确的 $\boldsymbol{R}$, 其中 $\boldsymbol{Z}$ 是某一精确的正交矩阵且 $\|\boldsymbol{E}\|_2\approx\mathbf{u}\|\boldsymbol{A}\|_2$.

### 5.2.2 分块 Householder QR 分解

算法 5.2.1 含有大量的矩阵与向量相乘和外积修正等 2 级运算. 通过重新组织计算和应用 5.1.7 节中讨论的分块 Householder 表示形式可以得到一个 3 级运算方法. 其思想是应用一组如 5.1.7 节中的 WY 形式表示的 Householder 变换.

一个小例子可以说明主要思想. 假定 $n=12$ 且分块参数 $r$ 的值为 $r=3$. 第一步是如算法 5.2.1 那样生成 Householder 矩阵 $\boldsymbol{H}_1,\boldsymbol{H}_2$ 和 $\boldsymbol{H}_3$. 然而不像算法 5.2.1 那样将 $\boldsymbol{H}_i$ 作用于整个 $\boldsymbol{A}$, 我们只将 $\boldsymbol{H}_1,\boldsymbol{H}_2$ 和 $\boldsymbol{H}_3$ 作用于 $\boldsymbol{A}(:,:3)$. 做完这些之后我们生成分块表示式 $\boldsymbol{H}_1\boldsymbol{H}_2\boldsymbol{H}_3=\boldsymbol{I}+\boldsymbol{W}_1\boldsymbol{Y}_1^{\mathrm{T}}$, 然后进行 3 级修正

$$A(:,4:12)=(\boldsymbol{I}+\boldsymbol{W}\boldsymbol{Y}^{\mathrm{T}})A(:,4:12).$$

下一步, 按算法 5.2.1 生成 $\boldsymbol{H}_4,\boldsymbol{H}_5$ 和 $\boldsymbol{H}_6$. 然而在得到分块表示形式 $\boldsymbol{H}_4\boldsymbol{H}_5\boldsymbol{H}_6=\boldsymbol{I}+\boldsymbol{W}_2\boldsymbol{Y}_2^{\mathrm{T}}$ 之后才将这些变换作用于 $A(:,7:12)$. 这就是整体构想.

$\lambda=1;k=0$

**while** $\lambda\leqslant n$

  $\tau=\min(\lambda+r-1,n);k=k+1$

  用算法 5.2.1 上三角化 $A(\lambda:m,\lambda:n)$

    得到 Householder 矩阵 $H_\lambda,\cdots,H_\tau$.  (5.2.1)

  用算法 5.1.2 得到块形式

    $I+W_kY_k^{\mathrm{T}}=H_\lambda,\cdots,H_\tau$

  $A(\lambda:m,\tau+1:n)=(I+W_kY_k^{\mathrm{T}})^{\mathrm{T}}A(\lambda:m,\tau+1:n)$

  $\lambda=\tau+1$

**end**

用于定义矩阵 $\boldsymbol{H}_\lambda,\cdots,\boldsymbol{H}_\tau$ 的 Householder 向量的零–非零结构表明 $\boldsymbol{W}_k$ 和 $\boldsymbol{Y}_k$ 的前 $\lambda-1$ 行都是零. 该事实在实际应用中可利用.

考虑 (5.2.1) 的合适的方式是进行划分:

$$\boldsymbol{A}=[\boldsymbol{A}_1,\cdots,\boldsymbol{A}_N],\quad N=\mathrm{ceil}(n/r),$$

其中列块 $A_k$ 在第 $k$ 步被处理. 在 (5.2.1) 的第 $k$ 步, 一个分块 Householder 形成且将 $A_k$ 的次对角线部分化为 0. 余下的列块也被更新.

(5.2.1) 的舍入特性基本上与算法 5.2.1 相同. 由于 $W$ 矩阵的计算, 所需的 flop 数稍有增长. 然而作为分块的结果, 除一小部分外所有的 flop 都发生在矩阵乘法中. 确切地说, (5.2.1) 的 3 级比例约为 $1-2/N$. 详细内容请参阅 Bischof and Van Loan(1987).

### 5.2.3 Givens QR 方法

Givens 旋转也可用来计算 QR 分解. 用一个 $4\times 3$ 矩阵的例子足以表明其一般思想:

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{(3,4)} \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{(1,2)}$$

$$\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{(3,4)} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{(3,4)} \boldsymbol{R}.$$

这里我们已标记定义所对应 Givens 旋转的 2 向量. 很显然, 如果 $\boldsymbol{G}_j$ 代表在约化过程中的第 $j$ 次 Givens 旋转, 则 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{R}$ 是上三角形矩阵, 其中 $\boldsymbol{Q}=\boldsymbol{G}_1\cdots\boldsymbol{G}_t$. 且 $t$ 是旋转的总次数. 对于一般的 $m$ 和 $n$ 我们有下列算法.

**算法 5.2.2 (Givens QR)** 给定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 且 $m\geqslant n$, 本算法以 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{R}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{R}$ 是上三角形矩阵, $\boldsymbol{Q}$ 是正交矩阵.

**for** $j=1:n$

  **for** $i=m:-1:j+1$

    $[c,s]=\mathbf{givens}(A(i-1,j),A(i,j))$

$$A(i-1:i,j:n)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} A(i-1:i,j:n)$$

  **end**

**end**

本算法需要 $3n^2(m-n/3)$ 个 flop. 注意, 我们可利用 (5.1.9) 将 $(c,s)$ 通过单个 $\rho$ 来表示, 且将其存储于已化为零的 $A(i,j)$ 中. 利用 (5.1.10) 可以进行如 $\boldsymbol{x}\leftarrow\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x}$ 的运算, 但要仔细重组这些旋转的顺序.

对上三角形矩阵可采用其他顺序的旋转变换. 例如, 如果我们用下面的语句取代算法 5.2.2 中的for语句

**for** $i=m:-1:2$

  **for** $j=1:\min\{i-1,n\}$

则 $\boldsymbol{A}$ 中的零元是逐行引进的.

Givens QR 方法中的另一个因素是将 $a_{ij}$ 消零的旋转平面. 例如, 为使 $a_{ij}$ 变为 0, 如不像算法 5.2.2 中那样对第 $i-1$ 行和第 $i$ 行做旋转, 我们也可对第 $j$ 行和第 $i$ 行进行:

**for** $j=1:n$

```
        for i = m : -1 : j+1
            [c, s] = givens(A(j, j), A(i, j))
```
$$A([j\quad i], j:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} A([j\quad i], j:n)$$
```
        end
    end
```

### 5.2.4 利用 Givens 变换的 Hessenberg QR 分解

作为 Givens 旋转应用于有结构的问题的例子, 我们说明如何将其用于计算上 Housenberg 矩阵之 QR 分解. 一个小例子可以表明其一般思想.

假定 $n = 6$, 且经过两步变换后, 我们计算得

$$G(2,3,\theta_2)^{\mathrm{T}}G(1,2,\theta_1)^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

然后计算 $G(3,4,\theta_3)$ 来将当前的 (4, 3) 变为 0, 于是得到

$$G(3,4,\theta_3)^{\mathrm{T}}G(2,3,\theta_2)^{\mathrm{T}}G(1,2,\theta_1)^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

综上我们得到下列算法.

**算法 5.2.3 (Hessenberg QR)** 如 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是上 Hessenberg 矩阵, 则本算法用 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{R}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{Q}$ 是正交矩阵, $\boldsymbol{R}$ 是上三角形矩阵, $\boldsymbol{Q} = \boldsymbol{G}_1 \cdots \boldsymbol{G}_{n-1}$ 是一组 Givens 旋转的乘积, 其中 $\boldsymbol{G}_j$ 形如 $\boldsymbol{G}_j = G(j, j+1, \theta_j)$.

```
    for j = 1 : n-1
        [c s] = givens(A(j, j), A(j+1, j))
```
$$A(j:j+1, j:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} A(j:j+1, j:n)$$
```
    end
```

本算法需要 $3n^2$ 个 flop.

### 5.2.5 快速 Givens QR 分解

我们可以利用 5.1.13 节描述的快速 Givens 变换来计算 $\boldsymbol{Q}$ 的 $(\boldsymbol{M},\boldsymbol{D})$ 表示形式. 具体地说, 如 $\boldsymbol{M}$ 是非奇异的, $\boldsymbol{D}$ 是对角矩阵, 使得 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{T}$ 是上三角形矩阵, $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}=\boldsymbol{D}$ 是对角形矩阵, 则 $\boldsymbol{Q}=\boldsymbol{M}\boldsymbol{D}^{-1/2}$ 是正交矩阵, 且 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{D}^{-1/2}\boldsymbol{T}=\boldsymbol{R}$ 是上三角形矩阵. 类似于 Givens QR 方法我们有下述算法.

**算法 5.2.4 (快速 Givens QR)** 给定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 且 $m\geqslant n$, 本算法计算非奇异的 $\boldsymbol{M}\in\mathbb{R}^{m\times m}$ 和正的 $d(1:m)$ 使得 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{T}$ 为上三角形矩阵, 且 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}=\mathrm{diag}(d_1,\cdots,d_m)$. $\boldsymbol{A}$ 被 $\boldsymbol{T}$ 覆盖. 注意 $\boldsymbol{A}=(\boldsymbol{M}\boldsymbol{D}^{-1/2})(\boldsymbol{D}^{1/2}\boldsymbol{T})$ 是 $\boldsymbol{A}$ 的 QR 分解.

**for** $i=1:m$
  $d(i)=1$
**end**
**for** $j=1:n$
  **for** $i=m:-1:j+1$
    $[\alpha,\beta,\mathrm{type}]=\mathbf{fast.givens}(A(i-1:i,j),d(i-1:i))$
    **if** type $=1$

$$A(i-1:i,j:n)=\begin{bmatrix}\beta & 1\\ 1 & \alpha\end{bmatrix}^{\mathrm{T}}A(i-1:i,j:n)$$

    **else**

$$A(i-1:i,j:n)=\begin{bmatrix}1 & \alpha\\ \beta & 1\end{bmatrix}^{\mathrm{T}}A(i-1:i,j:n)$$

  **end**
**end**

本算法需要 $2n^2(m-n/3)$ 个 flop. 正如上一节所提到的, 有必要防止上述的快速 Givens 算法出现溢出. 这意味着如果 $\boldsymbol{M},\boldsymbol{D}$ 和 $\boldsymbol{A}$ 的元素变得很大时要定期对它们加权缩小.

如果要对一个窄带状矩阵做 QR 分解, 则快速 Givens 方法就很有吸引力, 因为它不涉及平方根运算. (在窄带状矩阵问题上 $\mathrm{LDL}^{\mathrm{T}}$ 比 Cholesky 分解更令人喜欢, 这出于相同的原因, 参见 4.3.6 节). 具体地说, 如果 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 的上带宽为 $q$, 下带宽为 $p$, 则 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{R}$ 的上带宽为 $p+q$. 在这种情况下, Givens QR 需大约 $O(np(p+q))$ 个 flop 和 $O(np)$ 次平方根运算. 因此, 如 $p,q\ll n$, 平方根运算在整个计算中占相当大的比例.

### 5.2.6 QR 分解的性质

上面的算法"证明"了 QR 分解的存在. 现在我们讨论 $\boldsymbol{Q}$ 的列向量与 $\mathrm{ran}(\boldsymbol{A})$ 和 $\mathrm{ran}(\boldsymbol{A})^{\perp}$ 的关系以及考察 QR 分解的唯一性问题.

**定理 5.2.1** 如果 $A = QR$ 是一个列满秩阵 $A \in \mathbb{R}^{m\times n}$ 的 QR 分解, 且 $A = [a_1, \cdots, a_n], Q = [q_1, \cdots, q_m]$ 是一个按列的划分, 则

$$\text{span}\{a_1, \cdots, a_k\} = \text{span}\{q_1, \cdots, q_k\}, \quad k = 1:n.$$

确切地说, 如果 $Q_1 = Q(1:m, 1:n)$ 和 $Q_2 = Q(1:m, n+1:m)$, 则

$$\text{ran}(A) = \text{ran}(Q_1), \quad \text{ran}(A)^{\perp} = \text{ran}(Q_2)$$

且有 $A = Q_1R_1$, 其中 $R_1 = R(1:n, 1:n)$.

**证明** 比较 $A = QR$ 的第 $k$ 列, 我们有

$$a_k = \sum_{i=1}^{k} r_{ik}q_i \in \text{span}\{q_1, \cdots, q_k\}. \tag{5.2.2}$$

因此, $\text{span}\{a_1,\cdots,a_k\} \subseteq \text{span}\{q_1, \cdots, q_k\}$. 然而, 由 $\text{rank}(A) = n$ 推知 $\text{span}\{a_1,\cdots,a_k\}$ 的维数为 $k$, 故必等于 $\text{span}\{q_1, \cdots, q_k\}$, 定理的其余部分显然成立. □

矩阵 $Q_1 = Q(1:m, 1:n)$ 和 $Q_2 = Q(1:m, n+1:m)$ 可用 $Q$ 的因子形式很容易地求出.

如果 $A = QR$ 是 $A$ 的 QR 分解, 其中 $A \in \mathbb{R}^{m\times n}$, 且 $m \geqslant n$, 则我们称 $A = Q(:, 1:n)R(1:n, 1:n)$ 为窄 QR 分解, 下面的结论给出了窄 QR 分解的唯一性.

**定理 5.2.2** 假定 $A \in \mathbb{R}^{m\times n}$ 是列满秩的, 则窄 QR 分解

$$A = Q_1R_1$$

是唯一的, 其中 $Q_1 \in \mathbb{R}^{m\times n}$ 的列向量相互正交, $R_1$ 是对角元素大于 0 的上三角形矩阵. 而且 $R_1 = G^{\mathrm{T}}$, 其中 $G$ 是 $A^{\mathrm{T}}A$ 的下三角 Cholesky 因子.

**证明** 由 $A^{\mathrm{T}}A = (Q_1R_1)^{\mathrm{T}}(Q_1R_1) = R_1^{\mathrm{T}}R_1$ 可以看出 $G = R_1^{\mathrm{T}}$ 是 $A^{\mathrm{T}}A$ 的 Cholesky 因子. 从定理 4.2.5 可知此因子是唯一的. 由于 $Q_1 = AR_1^{-1}$, 故 $Q_1$ 也是唯一的. □

$A$ 中的扰动是如何影响 $Q_1$ 和 $R_1$ 的呢? 为回答这个问题, 我们需要将条件数推广到矩阵. 回想在 2.7.3 节中, 一个非奇异方阵的 2 范数的条件数是最大奇异值和最小奇异值的比. 对于列满秩的矩阵, 我们继续用这样的定义:

$$A \in \mathbb{R}^{m\times n}, \quad \text{rank}(A) = n \Rightarrow \kappa_2(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}.$$

如果 $A$ 的列是接近相关, 则 $\kappa_2(A)$ 会很大. Stewart(1993) 证明了 $A$ 中 $O(\varepsilon)$ 的相对误差会导致 $R$ 和 $Q_1$ 中 $O(\varepsilon\kappa_2(A))$ 的相对误差.

### 5.2.7 经典 Gram-Schmidt 算法

现在我们讨论直接计算 QR 分解 $A = Q_1R_1$ 的两种不同方法. 如果 $\text{rank}(A) = n$, 则从方程 (5.2.2) 可解出 $q_k$:

$$q_k = \left(a_k - \sum_{i=1}^{k-1} r_{ik}q_i\right) \Big/ r_{kk}.$$

这样我们可以认为 $q_k$ 是在

$$z_k = a_k - \sum_{i=1}^{k-1} r_{ik} q_i$$

方向上的单位 2 范数向量. 为保证 $z_k \in \text{span}\{q_1, \cdots, q_{k-1}\}^\perp$, 我们选取

$$r_{ik} = q_i^{\mathrm{T}} a_k, \quad i = 1 : k-1.$$

由此导出解 $\boldsymbol{A} = \boldsymbol{Q}_1 \boldsymbol{R}_1$ 的经典 Gram-Schmidt(CGS) 算法:

$$\begin{array}{l}
R(1,1) = \|A(:,1)\|_2 \\
Q(:,1) = A(:,1)/R(1,1) \\
\textbf{for } k = 2:n \\
\qquad R(1:k-1,k) = Q(1:m,1:k-1)^{\mathrm{T}} A(1:m,k) \\
\qquad z = A(1:m,k) - Q(1:m,1:k-1)R(1:k-1,k) \\
\qquad R(k,k) = \|z\|_2 \\
\qquad Q(1:m,k) = z/R(k,k) \\
\textbf{end}
\end{array} \tag{5.2.3}$$

在 CGS 的第 $k$ 步, 生成 $\boldsymbol{Q}$ 和 $\boldsymbol{R}$ 的第 $k$ 列.

### 5.2.8 修正 Gram-Schmidt 算法

可惜, CGS 的数值特性非常坏, 因为所计算的 $q_i$ 之间的正交性常常会严重损失. 有意思的是, 改变计算的次序, 便得到所谓的修正 Gram-Schmidt(MGS), 这是一个可靠得多的计算方法. 在 MGS 的第 $k$ 步, 需要求出 $\boldsymbol{Q}$ 的第 $k$ 列 (用 $q_k$ 表示) 和 $\boldsymbol{R}$ 的第 $k$ 行 (用 $r_k^{\mathrm{T}}$ 表示). 为导出 MGS 方法, 定义矩阵 $\boldsymbol{A}^{(k)} \in \mathbb{R}^{m \times (n-k+1)}$ 为

$$\boldsymbol{A} - \sum_{i=1}^{k-1} q_i r_i^{\mathrm{T}} = \sum_{i=k}^{n} q_i r_i^{\mathrm{T}} = [0 \quad A^{(k)}]. \tag{5.2.4}$$

因此, 如果

$$\boldsymbol{A}^{(k)} = \left[ \begin{array}{cc} z & \boldsymbol{B} \\ 1 & n-k \end{array} \right],$$

则 $r_{kk} = \|z\|_2, q_k = z/r_{kk}$ 以及 $(r_{k,k+1} \cdots r_{kn}) = q_k^{\mathrm{T}} \boldsymbol{B}$. 然后计算外积 $\boldsymbol{A}^{(k+1)} = \boldsymbol{B} - q_k(r_{k,k+1} \cdots r_{kn})$ 并开始下一步. 至此, 我们完成了 MGS 第 $k$ 步的描述.

**算法 5.2.5 (修正 Gram-Schmidt)** 给定 $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, $\text{rank}(\boldsymbol{A}) = n$, 本算法计算分解 $\boldsymbol{A} = \boldsymbol{Q}_1 \boldsymbol{R}_1$, 其中 $\boldsymbol{Q}_1 \in \mathbb{R}^{m \times n}$, 其列相互正交, $\boldsymbol{R}_1 \in \mathbb{R}^{n \times n}$ 是上三角形矩阵.

$$\begin{array}{l}
\textbf{for } k = 1:n \\
\qquad R(k,k) = \|A(1:m,k)\|_2 \\
\qquad Q(1:m,k) = A(1:m,k)/R(k,k) \\
\qquad \textbf{for } j = k+1:n \\
\qquad\qquad R(k,j) = Q(1:m,k)^{\mathrm{T}} A(1:m,j) \\
\qquad\qquad A(1:m,j) = A(1:m,j) - Q(1:m,k)R(k,j)
\end{array}$$

```
    end
end
```

本算法需 $2mn^2$ 个 flop. 把 $\boldsymbol{Q}_1$ 和 $\boldsymbol{R}_1$ 都储存于 $\boldsymbol{A}$ 是不可能的. MGS 计算的典型安排是使 $\boldsymbol{A}$ 被 $\boldsymbol{Q}_1$ 覆盖, 矩阵 $\boldsymbol{R}_1$ 储存在另外的数组中.

### 5.2.9 工作量和精度

如果对求解 ran($\boldsymbol{A}$) 的正交基感兴趣, 那么用 Householder 方法产生因子形式的 $\boldsymbol{Q}$ 需 $2mn^2 - 2n^3/3$ 个 flop, 计算 $\boldsymbol{Q}$ 的前 $n$ 列还需要 $2mn^2 - 2n^3/3$ 个 flop. (这只需注意 (5.1.5) 中 $\boldsymbol{Q}$ 的前 $n$ 列.) 因此, 对于找 ran($\boldsymbol{A}$) 的正交基, MGS 的效率要比 Householder 正交化高一倍. 但是, Björck(1967) 证明了 MGS 计算的 $\hat{\boldsymbol{Q}}_1 = [\hat{q}_1, \cdots, \hat{q}_n]$ 满足

$$\hat{\boldsymbol{Q}}_1^{\mathrm{T}}\hat{\boldsymbol{Q}}_1 = \boldsymbol{I} + \boldsymbol{E}_{\mathrm{MGS}}, \quad \|\boldsymbol{E}_{\mathrm{MGS}}\|_2 \approx \mathbf{u}\kappa_2(\boldsymbol{A}),$$

而 Householder 方法计算的结果是

$$\hat{\boldsymbol{Q}}_1^{\mathrm{T}}\hat{\boldsymbol{Q}}_1 = \boldsymbol{I} + \boldsymbol{E}_H, \quad \|\boldsymbol{E}_H\|_2 \approx \mathbf{u}.$$

因此, 如果正交性至关重要, 则仅当被正交化的向量独立性强的时候, 才可用 MGS 求正交基.

同时我们指出由 MGS 计算出的三角因子 $\hat{\boldsymbol{R}}$ 满足 $\|\boldsymbol{A} - \hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}\| \approx \mathbf{u}\|\boldsymbol{A}\|$, 且存在一个具有完全正交列向量的 $\boldsymbol{Q}$ 使得 $\|\boldsymbol{A} - \boldsymbol{Q}\hat{\boldsymbol{R}}\| \approx \mathbf{u}\|\boldsymbol{A}\|$. 参阅 Higham(1996, 第 379 页).

**例 5.2.1** *如果将修正 Gram-Schmidt 应用于*

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 \\ 10^{-3} & 0 \\ 0 & 10^{-3} \end{bmatrix}, \quad \kappa_2(\boldsymbol{A}) \approx 1.4 \times 10^3,$$

采用 6 位有效数计算, 则

$$[\hat{q}_1 \quad \hat{q}_2] = \begin{bmatrix} 1.000\,00 & 0 \\ 0.001 & -0.707\,107 \\ 0 & 0.707\,100 \end{bmatrix}.$$

### 5.2.10 关于复矩阵 QR 分解的一点说明

本书中我们给出的大部分算法都有其适用于复矩阵的形式, 它们可从相应的基于实数的算法直接得到. (但这并不等于说在实现层次上一切都非常容易和显而易见). 为表明这一点, 我们简略给出复矩阵的 QR 分解算法.

首先讨论 Householder 变换的情形, 假定 $\mathbf{0} \neq \boldsymbol{x} \in \mathbb{C}^n$ 且 $x_1 = re^{\mathrm{i}\theta}$, 其中 $r, \theta \in \mathbb{R}$. 如果 $\boldsymbol{v} = \boldsymbol{x} \pm \mathrm{e}^{\mathrm{i}\theta}\|\boldsymbol{x}\|_2 e_1, \boldsymbol{P} = \boldsymbol{I}_n - \beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{H}}, \beta = 2/\boldsymbol{v}^{\mathrm{H}}\boldsymbol{v}$, 则 $\boldsymbol{P}x = \mp\mathrm{e}^{\mathrm{i}\theta}\|\boldsymbol{x}\|_2 e_1$ (参见习题 5.1.3). 为了保证计算的稳定性, 符号的选取应使 $\|\boldsymbol{v}\|_2$ 极大化.

对 $\boldsymbol{A} \in \mathbb{C}^{m\times n}, m \geqslant n$ 进行上三角形化, 按算法 5.2.1 进行. 在第 $j$ 步, 将 $A(j:m,j)$ 的次对角线部分消为 0:

**for** $j=1:n$

$x = A(j:m,j)$

$v = x \pm \mathrm{e}^{\mathrm{i}\theta}\|x\|_2 e_1$, 这里 $x_1 = r\mathrm{e}^{\mathrm{i}\theta}$

$\beta = 2/v^{\mathrm{H}}/v$

$A(j:m,j:n) = (I_{m-j+1} - \beta vv^{\mathrm{H}})A(j:m,j:n)$

**end**

此约化需 $8n^2(m-n/3)$ 个实 flop, 这大约为执行算法 5.2.1 的四倍. 如 $\boldsymbol{Q} = \boldsymbol{P}_1\cdots\boldsymbol{P}_n$ 是 Householder 变换的乘积, 则 $\boldsymbol{Q}$ 是酉矩阵且 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{R} \in \mathbb{C}^{m\times n}$ 是复上三角形矩阵.

## 习　题

**5.2.1**　改写 Householder QR 算法, 使得它能有效地处理 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 是下带宽为 $p$, 上带宽为 $q$ 的情形.

**5.2.2**　改写 Householder QR 算法, 使其计算分解 $\boldsymbol{A} = \boldsymbol{QL}$, 其中 $\boldsymbol{L}$ 是下三角形矩阵, $\boldsymbol{Q}$ 是正交矩阵. 假定 $\boldsymbol{A}$ 是方阵. 这涉及重写 Householder 向量函数 $\boldsymbol{v} = \mathbf{house}(\boldsymbol{x})$ 使得 $(\boldsymbol{I} - 2\boldsymbol{vv}^{\mathrm{T}}/\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v})\boldsymbol{x}$ 除最后一个分量外均为 0.

**5.2.3**　改写 Givens QR 分解算法使得在对角线上引入零元, 即按照顺序将 $(m,1),(m-1,1),(m,2),\ (m-2,1),(m-1,2),(m,3)$ 等元素变为 0.

**5.2.4**　改写快速 Givens QR 分解算法使得它能有效处理 $\boldsymbol{A}$ 是 $n\times n$ 三对角矩阵的情形, 假定 $\boldsymbol{A}$ 的次对角线、对角线、超对角线分别储存于 $e(1:n-1), a(1:n), f(1:n-1)$ 中. 设计你的算法, 使这些向量被 $T$ 的非零部分覆盖.

**5.2.5**　假设 $\boldsymbol{L} \in \mathbb{R}^{m\times n}$ $(m \geqslant n)$ 是下三角形矩阵, 证明 Householder 矩阵 $\boldsymbol{H}_1\cdots\boldsymbol{H}_n$ 可以确定一个下三角形矩阵 $\boldsymbol{L}_1 \in \mathbb{R}^{n\times n}$, 满足

$$\boldsymbol{H}_n\cdots\boldsymbol{H}_1\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_1 \\ 0 \end{bmatrix}.$$

提示: $6\times 3$ 矩阵的第二步涉及确定 $\boldsymbol{H}_2$ 使得

$$\boldsymbol{H}_2\begin{bmatrix} \times & 0 & 0 \\ \times & \times & 0 \\ \times & \times & \times \\ \times & \times & 0 \\ \times & \times & 0 \\ \times & \times & 0 \end{bmatrix} = \begin{bmatrix} \times & 0 & 0 \\ \times & \times & 0 \\ \times & \times & \times \\ \times & 0 & 0 \\ \times & 0 & 0 \\ \times & 0 & 0 \end{bmatrix},$$

即只有第一行和第三行保持原样.

**5.2.6**　证明, 如果

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{w} \\ \boldsymbol{0} & \boldsymbol{v} \end{bmatrix}\begin{matrix} k \\ m-k \end{matrix} \qquad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}\begin{matrix} k \\ m-k \end{matrix}$$
$$\qquad k \qquad n-k$$

且 $A$ 是列满秩的, 则 $\min\|Ax-b\|_2^2=\|d\|_2^2-(v^{\mathrm{T}}d/\|v\|_2)^2$.

**5.2.7** 假定 $A\in\mathbb{R}^{n\times n}$ 和 $D=\mathrm{diag}(d_1,\cdots,d_n)\in\mathbb{R}^{n\times n}$, 给出一个方法来构造一个正交矩阵 $Q$ 使得 $Q^{\mathrm{T}}A-DQ^{\mathrm{T}}=R$ 是上三角形矩阵. 不必考虑效率问题 —— 这只是 QR 分解的一个小练习.

**5.2.8** 给出一个方法计算乘积 $A=A_p\cdots A_2A_1$ 的 QR 分解, 但不显式地将 $A_1,\cdots,A_p$ 相乘. 提示: 在 $p=3$ 的情形, $Q_3^{\mathrm{T}}A=Q_3^{\mathrm{T}}A_3Q_2Q_2^{\mathrm{T}}A_2Q_1Q_1^{\mathrm{T}}A_1$, 然后确定正交矩阵 $Q_i$ 使得 $Q_i^{\mathrm{T}}(A_iQ_{i-1})$ 是上三角形矩阵 $(Q_0=I)$.

**5.2.9** 假定 $A\in\mathbb{R}^{n\times n}$, $E$ 是将单位矩阵 $I_n$ 的行逆序排列得到的置换矩阵 (即 4.7 节中提到的交换矩阵), (a) 证明如 $R\in\mathbb{R}^{n\times n}$ 是上三角形矩阵, 则 $L=ERE$ 是下三角形矩阵. (b) 指出如何计算正交矩阵 $Q\in\mathbb{R}^{n\times n}$ 和下三角形矩阵 $L\in\mathbb{R}^{n\times n}$ 使得 $A=QL$, 假定已有 QR 分解的算法.

**5.2.10** $A\in\mathbb{R}^{m\times n}$ 上的 MGS 与

$$\tilde{A}=\begin{bmatrix} O_n \\ A \end{bmatrix}$$

上的 Householder QR 的第一步在数值上是等价的, 其中 $O_n$ 是 $n\times n$ 的零矩阵. 证明以上两种方法的第一步执行完后上述命题成立.

**5.2.11** 逆转算法 5.2.5(MGS QR) 中循环的次序使得 $R$ 是按列计算的.

**5.2.12** 写一个复矩阵的 Givens QR 分解方法, 参考习题 5.1.5, 其中复矩阵 Givens 旋转是主题. 能否组织计算过程使得 $R$ 的对角元非负?

## 本节注释与参考文献

应用 Householder 变换来求解 LS 问题的思想由下文提出:

A. S. Householder(1958). "Unitary Triangularization of a Nonsymmetric Matrix," *J. ACM.* 5, 339–342.

有实施的细节见:

P. Businger and G. H. Golub(1965). "Linear Least Squares Solutions by Householder Transformations," *Numer. Math.* 7, 269–276, See also Wilkinson and Reinsch (1971, 111–118.)

G. H. Golub(1965). "Numerical Methods for Solving Linear Least Squares Problems," *Numer. Math* 7, 206–216.

其于 Givens 旋转的 QR 分解之主要文献包括:

W. Givens(1958), "Computation of Plane Unitary Rotations Transforming a General Matrix to Triangular Form," *SIAM J. Appl. Math.* 6, 26–50.

M. Gentleman(1973). "Error Analysis of QR Decompositions by Givens Transformations," *Lin. Alg. and Its Appl.* 10, 189–197.

关于如何将 QR 分解应用于大量的统计计算问题中的讨论参见:

G. H. Golub(1969), "Matrix Decompositions and Statistical Computation," in *Statistical Computation,* ed. R. C. Milton and J. A. Nelder, Academic Press, New York, pp. 365–397

当 $A$ 有扰动时, $Q$ 和 $R$ 的变化在下文中讨论:

G. W. Stewart(1977), "Perturbation Bounds for the QR Factorization of a Matrix," *SIAM J. Num. Anal.* 14, 509–518.

H. Zha(1993). "A Componentwise Perturbation Analysis of the QR Decomposition," *SIAM J. Matrix Anal. Appl.* 4, 1124–1131.

G. W. Stewart (1993). "On the Perturbation of LU Cholesky, and QR Factorizations," *SIAM J. Matrix Anal. Appl.* 14, 1141–1145.

A. Barrlund(1994). "Perturbation Bounds for the Generalized QR Factorization," *Lin, Alg. and Its Applic.* 207, 251–271.

J. -G Sun(1995). "On Perturbation Bounds for the QR Factorization," *Lin. Alg. and Its Applic.* 215, 95–112.

最主要的结论是: $Q$ 和 $R$ 的变化之界是 $A$ 的条件数与 $A$ 的相对变化量的乘积. 组织计算过程可使得 $Q$ 的元素连续地依赖 $A$ 的元素, 此问题的讨论见:

T. F. Coleman and D. C. Sorensen(1984), "A Note on the Computation of an Orthonormal Basis for the Null Space of a Matrix," *Mathematical Programming* 29, 234–242.

有关 Gram-Schmidt 过程的参考文献包括:

J. R. Rice(1966). "Experiments on Gram-Schmidt Orthogonalization," *Math. Comp.* 20, 325–328.

A. Björck(1967). "Solving Linear Least Squares Problems by Gram-Schmidt Orthogonalization," *BIT* 7, 1–21.

N. N. Abdelmalek(1971). "Roundoff Error Analysis for Gram-Schmidt Method and Solution of Linear Least Squares Problems," *BIT* 11, 345–368.

J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart (1976). "Reorthogonalization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization," *Math. Comp.* 30, 772–795.

A. Ruhe(1983). "Numerical Aspects of Gram-Schmidt Orthogonalization of Vectors," *Lin. Alg. and Its Applic.* 52/53, 591–601.

W. Jalby and B. Philippe(1991). "Stability Analysis and Improvement of the Block Gram-Schmidt Algorithm," *SIAM J. Sci, Stat. Comp.* 12, 1058–1073.

Å. Björck and C. C. Paige (1992). "Loss and Recapture of Orthogonality in the Modified Gram-Schmidt Algorithm," *SIAM J. Matrix Anal. Appl.* 13, 176–190.

A. Björck(1994). "Numerics of Gram-Schmidt Orthogonalization," *Lin. Alg. and Its Applic,* 197/198, 297–316.

结构化的矩阵之 QR 分解本身也是结构化的, 参阅:

A. W. Bojanczyk, R. P. Brent, and F. R. de Hoog (1986). "QR Factorization of Toeplitz Matrices," *Numer. Math.* 49, 81–94.

S. Qiao (1986). "Hybrid Algorithm for Fast Toeplitz Orthogonalization," *Numer. Math.* 53, 351–366.

C. J. Demeure(1989). "Fast QR Factorization of Vandermonde Matrices," *Lin. Alg. and Its Applic.* 122/123/124, 165–194.

L. Reichel(1991), "Fast QR Decomposition of Vandermonde-Like Matrices and Polynomial Least Squares Approximation," *SIAM J. Matrix Anal. Appl.* 12, 552–564.

D. R. Sweet (1991). "Fast Block Toeplitz Orthogonalization," *Numer. Math.* 58, 613–629.

与 QR 分解相关的许多高性能计算问题之讨论可见：

B. Mattingly, C. Meyer, and J. Ortega(1989). "Orthogonal Reduction on Vector Computers," *SIAM J. Sci, and Stat. Comp.* 10, 372–381.

P. A. Knight(1995). "Fast Rectangular Matrix Multiplication and the QR Decomposition," *Lin. Alg. and Its Applic.* 221, 69–81.

## 5.3 满秩的 LS 问题

考虑如下问题：找到向量 $\boldsymbol{x} \in \mathbb{R}^n$ 使得 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, 其中数据矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 和观察向量 $\boldsymbol{b} \in \mathbb{R}^m$ 给定, $m \geqslant n$. 如果方程个数多于未知量个数, 我们称方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 是超定的. 超定方程组通常没有精确解, 因为它要求 $\boldsymbol{b}$ 必须是 $\mathbb{R}^m$ 的真子空间 $\mathrm{ran}(\boldsymbol{A})$ 的一个元素.

这一事实启发了我们, 可以考虑对适当选取的 $p$, 极小化 $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_p$. 不同的范数给出不同的最优解. 例如 $\boldsymbol{A} = [1,1,1]^{\mathrm{T}}, \boldsymbol{b} = [b_1, b_2, b_3]^{\mathrm{T}}$, 并且 $b_1 \geqslant b_2 \geqslant b_3 \geqslant 0$, 则可以验证：

$$\begin{aligned} p = 1 &\Rightarrow x_{opt} = b_2, \\ p = 2 &\Rightarrow x_{opt} = (b_1 + b_2 + b_3)/3, \\ p = \infty &\Rightarrow x_{opt} = (b_1 + b_3)/2. \end{aligned}$$

在 1 范数和 $\infty$ 范数情况下, 极小化工作变复杂了, 因为对于这些 $p$ 值, 函数 $f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_p$ 是不可微的. 但是在这方面已取得长足的进步, 已有一些好的方法可用于 1 范数和 $\infty$ 范数的极小化. 参阅 Coleman and Li(1992), Li(1993) 和 Zhang(1993).

与一般的 $p$ 范数极小化相比, **最小二乘**(LS) 问题

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 \tag{5.3.1}$$

更容易, 这由于以下两个原因：

- $\phi(\boldsymbol{x}) = \dfrac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$ 是 $x$ 的可微函数, 故 $\phi$ 取极小值时, 梯度方程 $\nabla\phi(\boldsymbol{x}) = 0$, 这是一个很容易有结构的对称线性方程组, 当 $\boldsymbol{A}$ 是列满秩时, 它是正定的.
- 2 范数在正交变换下不变, 这意味着我们可以寻找一正交矩阵 $\boldsymbol{Q}$ 使得等价的问题 $\|(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{x} - (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b})\|_2$ 的最小化容易求解.

本节中在我们致力于寻求在 $\boldsymbol{A}$ 为列满秩时这两种求解方法. 我们将详细给出并比较基于法方程的方法和基于 QR 分解的方法.

### 5.3.1 满秩的实质

假定 $\boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{z} \in \mathbb{R}^n$, 和 $\alpha \in \mathbb{R}$, 考虑等式

$$\|A(x+\alpha z)-b\|_2^2 = \|Ax-b\|_2^2 + 2\alpha z^{\mathrm{T}} A^{\mathrm{T}}(Ax-b) + \alpha^2\|Az\|_2^2,$$

其中 $A \in \mathbb{R}^{m\times n}$ 和 $b \in \mathbb{R}^m$. 如果 $x$ 是 LS 问题 (5.3.1) 的解, 则必有 $A^{\mathrm{T}}(Ax-b)=0$. 否则, 如果 $z=-A^{\mathrm{T}}(Ax-b)$, 且使 $\alpha$ 足够小, 则会得到一个相矛盾的不等式 $\|A(x+\alpha z)-b\|_2 < \|Ax-b\|_2$. 我们还可得出结论: 如 $x$ 和 $x+\alpha z$ 都是 LS 的极小解, 则 $z \in \mathrm{null}(A)$.

因此, 如果 $A$ 是列满秩的, 则存在一个唯一的 LS 解 $x_{\mathrm{LS}}$, 它是对称正定线性方程组

$$A^{\mathrm{T}}Ax_{\mathrm{LS}} = A^{\mathrm{T}}b$$

的解. 这方程称为**法方程组**. 由于 $\nabla\phi(x) = A^{\mathrm{T}}(Ax-b)$, 其中 $\phi(x)=\frac{1}{2}\|Ax-b\|_2^2$, 因此求解法方程组等同于求解梯度方程 $\nabla\phi=0$. 我们称

$$r_{\mathrm{LS}} = b - Ax_{\mathrm{LS}}$$

为极小剩余, 用

$$\rho_{\mathrm{LS}} = \|Ax_{\mathrm{LS}} - b\|_2$$

表示其大小. 注意, 如果 $\rho_{\mathrm{LS}}$ 很小, 则我们可从 $A$ 的列 "预测" 出 $b$.

到目前为止我们一直假设 $A \in \mathbb{R}^{m\times n}$ 是列满秩的, 该假设在 5.5 节将去掉. 然而, 即使 $\mathrm{rank}(A)=n$, 当 $A$ 接近秩亏时, 上面给出的方法也会产生麻烦.

当评估一个 LS 计算解 $\hat{x}_{\mathrm{LS}}$ 的质量时, 应考虑两方面问题:

- $\hat{x}_{\mathrm{LS}}$ 有多靠近 $x_{\mathrm{LS}}$?
- 与 $r_{\mathrm{LS}} = b - Ax_{\mathrm{LS}}$ 相比, $\hat{r}_{\mathrm{LS}} = b - A\hat{x}_{\mathrm{LS}}$ 有多小?

在不同的应用中这两条标准的重要性也不尽相同. 但任何情况下都应知道 $A$ 和 $b$ 的扰动是如何影响 $x_{\mathrm{LS}}$ 和 $r_{\mathrm{LS}}$ 的. 直觉告诉我们, 如果 $A$ 的列是接近相关的, 那这些量会很灵敏.

**例 5.3.1**　假定

$$A=\begin{bmatrix}1 & 0\\ 0 & 10^{-6}\\ 0 & 0\end{bmatrix},\quad \delta A=\begin{bmatrix}0 & 0\\ 0 & 0\\ 0 & 10^{-8}\end{bmatrix},\quad b=\begin{bmatrix}1\\0\\1\end{bmatrix},\quad \delta b=\begin{bmatrix}0\\0\\0\end{bmatrix},$$

$x_{\mathrm{LS}}$ 和 $\hat{x}_{\mathrm{LS}}$ 分别使 $\|Ax-b\|_2$ 和 $\|(A+\delta A)x-(b+\delta b)\|_2$ 取得极小值, $r_{\mathrm{LS}}$ 和 $\hat{r}_{\mathrm{LS}}$ 分别是对应的极小余量, 则

$$x_{\mathrm{LS}}=\begin{bmatrix}1\\0\end{bmatrix},\quad \hat{x}_{\mathrm{LS}}=\begin{bmatrix}1\\ 0.9999\times 10^4\end{bmatrix},$$

$$r_{\mathrm{LS}}=\begin{bmatrix}0\\0\\1\end{bmatrix},\quad \hat{r}_{\mathrm{LS}}=\begin{bmatrix}0\\ -0.9999\times 10^{-2}\\ 0.9999\times 10^{0}\end{bmatrix}.$$

因为 $\kappa_2(A)=10^6$, 我们有

$$\frac{\|\hat{x}_{\mathrm{LS}}-x_{\mathrm{LS}}\|_2}{\|x_{\mathrm{LS}}\|_2} \approx 0.9999\times 10^4 \leqslant \kappa_2(A)^2\frac{\|\delta A\|_2}{\|A\|_2} = 10^{12}\times 10^{-8},$$

$$\frac{\|\hat{r}_{\rm LS}-r_{\rm LS}\|_2}{\|b\|_2}\approx 0.7070\times 10^{-2}\leqslant \kappa_2(\boldsymbol{A})\frac{\|\delta \boldsymbol{A}\|_2}{\|\boldsymbol{A}\|_2}=10^6\times 10^{-8}.$$

这个例子的启示是 $x_{\rm LS}$ 的灵敏度依赖于 $\kappa_2(\boldsymbol{A})^2$, 本节的最后我们将研究 LS 问题的扰动理论并会重新提到 $\kappa_2(\boldsymbol{A})^2$ 因子.

### 5.3.2 法方程组方法

求解列满秩的 LS 问题应用最广的方法是法方程组法.

**算法 5.3.1 (法方程组法)** 给定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 具有性质 $\operatorname{rank}(\boldsymbol{A})=n$ 和 $\boldsymbol{b}\in\mathbb{R}^m$, 本算法计算 LS 问题 $\min\|\boldsymbol{Ax}-\boldsymbol{b}\|_2$ 的解 $x_{\rm LS}$:

计算 $C=A^TA$ 的下三角部分

$d=A^{\rm T}b$.

计算 Cholesky 分解 $C=GG^{\rm T}$.

解 $Gy=d$ 和 $G^{\rm T}x_{\rm LS}=y$.

本算法需要 $(m+n/3)n^2$ 个 flop. 法方程组法基于许多标准算法: Cholesky 分解, 矩阵与矩阵相乘, 矩阵与向量相乘等, 因此很方便. 将 $m\times m$ 数据矩阵 $A$ 压缩于一个较小的 $n\times n$ 的叉积矩阵 $\boldsymbol{C}$ 的做法很可取.

下面我们考虑法方程组的计算解 $\hat{\boldsymbol{x}}_{\rm LS}$ 的精度. 为清晰起见, 假定在形成 $\boldsymbol{C}=\boldsymbol{A}^{\rm T}\boldsymbol{A}$ 和 $\boldsymbol{d}=\boldsymbol{A}^{\rm T}\boldsymbol{b}$ 时没有舍入误差 (这里的内积计算部分通常采用双精度的累加计算, 因而上述假定并非很不合理). 从我们所知的关于 Cholesky 分解的舍入性质 (见 4.2.7 节) 得到

$$(\boldsymbol{A}^{\rm T}\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{x}}_{\rm LS}=\boldsymbol{A}^{\rm T}\boldsymbol{b},$$

其中 $\|\boldsymbol{E}\|_2\approx \mathbf{u}\|\boldsymbol{A}^{\rm T}\|_2\|\boldsymbol{A}\|_2\approx \mathbf{u}\|\boldsymbol{A}^{\rm T}\boldsymbol{A}\|_2$, 因此我们期望

$$\frac{\|\hat{x}_{\rm LS}-x_{\rm LS}\|_2}{\|x_{\rm LS}\|_2}\approx \mathbf{u}\kappa_2(\boldsymbol{A}^{\rm T}\boldsymbol{A})=\mathbf{u}\kappa_2(\boldsymbol{A})^2. \tag{5.3.2}$$

可见, 法方程组计算解的精度依赖于条件数的平方. 这与例 5.3.1 是吻合的, 在 5.3.9 节中将给出更详细的结论.

**例 5.3.2** 要指出的是, 形成 $\boldsymbol{A}^{\rm T}\boldsymbol{A}$ 的过程会导致严重的信息丢失.

$$\boldsymbol{A}=\begin{bmatrix}1 & 1\\ 10^{-3} & 0\\ 0 & 10^{-3}\end{bmatrix},\quad \boldsymbol{b}=\begin{bmatrix}2\\ 10^{-3}\\ 10^{-3}\end{bmatrix},$$

则 $\kappa_2(\boldsymbol{A})\approx 1.4\times 10^3, \boldsymbol{x}_{\rm LS}=[1\quad 1]^{\rm T}$, 且 $\rho_{\rm LS}=0$. 如果采用以 10 为底, 长度 $t=6$ 的有限位数计算, 法方程组法就会在计算过程中产生除零现象, 这是由于

$$fl(\boldsymbol{A}^{\rm T}\boldsymbol{A})=\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$$

是奇异的. 另一方面, 如果采用 7 位数运算, 则 $\hat{\boldsymbol{x}}_{\rm LS}=[2.000\ 001,\ 0]^{\rm T}, \|\hat{\boldsymbol{x}}_{\rm LS}-\boldsymbol{x}_{\rm LS}\|_2/\|\boldsymbol{x}_{\rm LS}\|_2\approx \mathbf{u}\kappa_2(\boldsymbol{A})^2$.

### 5.3.3 用 QR 分解求解 LS 问题

设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, m \geqslant n$ 且 $\boldsymbol{b} \in \mathbb{R}^m$ 给定, 假定已计算得一个正交矩阵 $\boldsymbol{Q} \in \mathbb{R}^{m\times m}$ 满足

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{0} \end{bmatrix} \begin{matrix} n \\ m-n \end{matrix} \tag{5.3.3}$$

是上三角形矩阵. 如果

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix} \begin{matrix} n \\ m-n \end{matrix}$$

则对任意 $\boldsymbol{x} \in \mathbb{R}^n$ 有

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = \|\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b}\|_2^2 = \|\boldsymbol{R}_1\boldsymbol{x} - \boldsymbol{c}\|_2^2 + \|\boldsymbol{d}\|_2^2.$$

显然, 如果 $\mathrm{rank}(\boldsymbol{A}) = \mathrm{rank}(\boldsymbol{R}_1) = n$, 则 $\boldsymbol{x}_{\mathrm{LS}}$ 由上三角方程组 $\boldsymbol{R}_1\boldsymbol{x}_{\mathrm{LS}} = \boldsymbol{c}$ 定义. 注意到

$$\rho_{\mathrm{LS}} = \|\boldsymbol{d}\|_2.$$

我们看出, 一旦有了 $\boldsymbol{A}$ 的 QR 分解, 则满秩的 LS 问题就很容易求解. 细节依赖于精确的 QR 分解. 如果使用 Householder 矩阵且对 $\boldsymbol{b}$ 采用 $\boldsymbol{Q}^{\mathrm{T}}$ 的因子形式, 就得到如下算法.

**算法 5.3.2 (Householder LS 解法)** 如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 是列满秩的, 且 $\boldsymbol{b} \in \mathbb{R}^m$, 本算法计算向量 $\boldsymbol{x}_{\mathrm{LS}} \in \mathbb{R}^n$ 使得 $\|\boldsymbol{A}\boldsymbol{x}_{\mathrm{LS}} - \boldsymbol{b}\|_2$ 极小化.

利用算法 5.2.1 求 $\boldsymbol{A}$ 的 QR 分解并覆盖 $\boldsymbol{A}$.

**for** $j = 1:n$

  $v(j) = 1; v(j+1:m) = A(j+1:m, j)$

  $b(j:m) = (I_{m-j+1} - \beta_j vv^{\mathrm{T}})b(j:m)$

**end**

用向后消去法求解 $R(1:n, 1:n)x_{\mathrm{LS}} = b(1:n)$.

用这种方法求解满秩 LS 问题需要 $2n^2(m-n/3)$ 个 flop. 更新 $\boldsymbol{b}$ 所需的 $O(mn)$ 个 flop 和向后消去所需的 $O(n^2)$ 个 flop 与分解 $\boldsymbol{A}$ 的工作量相比是微不足道的.

可以证明, 计算得的 $\hat{\boldsymbol{x}}_{\mathrm{LS}}$ 是

$$\min \|(\boldsymbol{A} + \delta\boldsymbol{A})\boldsymbol{x} - (\boldsymbol{b} + \delta\boldsymbol{b})\|_2 \tag{5.3.4}$$

的解, 其中

$$\|\delta\boldsymbol{A}\|_F \leqslant (6m - 3n + 41)n\mathbf{u}\|\boldsymbol{A}\|_F + O(\mathbf{u}^2), \tag{5.3.5}$$

$$\|\delta\boldsymbol{b}\|_2 \leqslant (6m - 3n + 40)n\mathbf{u}\|\boldsymbol{b}\|_2 + O(\mathbf{u}^2). \tag{5.3.6}$$

这两个不等式是 Lawson and Hanson(1974, 90 页) 建立的, 并且指出 $\hat{\boldsymbol{x}}_{\mathrm{LS}}$ 满足一个 "附近" 的 LS 问题 (在建立 LS 扰动理论之前, 我们还不能讨论 $\hat{\boldsymbol{x}}_{\mathrm{LS}}$ 的相对误差, 该理论很快就要讨论). 要指出的是, 如果应用 Givens QR 分解, 也有类似的结果成立.

### 5.3.4 接近秩亏损时算法失败

如同法方程组法, 如果 $\text{rank}(\boldsymbol{A}) < n$, 则求解 LS 问题的 Householder 方法就会在向后消去阶段失败. 数值上, 只要 $\kappa_2(\boldsymbol{A}) = \kappa_2(\boldsymbol{R}) \approx 1/\mathbf{u}$ 就会出现麻烦. 而法方程组法, 一旦 $\kappa_2(\boldsymbol{A})$ 处于 $1/\sqrt{\mathbf{u}}$ 的邻域中, Cholesky 分解能否完成就可能出问题 (参见例 5.3.2). 因此 Lawson and Hanson(1974, 126~127 页) 宣称对于固定的机器精度, 应用 Householder 正交化方法可求解更多类型的 LS 问题.

### 5.3.5 MGS 方法的一点说明

实质上, MGS 方法计算窄 QR 分解 $\boldsymbol{A} = \boldsymbol{Q}_1\boldsymbol{R}_1$. 对于解列满秩的 LS 问题, 这就足够了, 因为它将法方程组 $(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{x} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$ 变换成上三角方程组 $\boldsymbol{R}_1\boldsymbol{x} = \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{b}$. 但当 $\boldsymbol{Q}_1^{\mathrm{T}}b$ 显式地形成时, 分析此方法却引出一个 $\kappa_2(\boldsymbol{A})^2$ 项. 这是由于所计算的因子 $\hat{\boldsymbol{Q}}_1$ 满足 $\|\hat{\boldsymbol{Q}}_1^{\mathrm{T}}\hat{\boldsymbol{Q}}_1 - \boldsymbol{I}_n\|_2 \approx \mathbf{u}\kappa_2(\boldsymbol{A})$, 这在 5.2.9 节中已提到.

然而, 如将 MGS 应用于增广矩阵

$$\boldsymbol{A}_+ = [\boldsymbol{A} \quad \boldsymbol{b}] = [\boldsymbol{Q}_1 \quad q_{n+1}]\begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{z} \\ \mathbf{0} & \rho \end{bmatrix},$$

则 $\boldsymbol{z} = \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{b}$. 按这种方法计算 $\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{b}$ 和解 $\boldsymbol{R}_1\boldsymbol{x}_{\mathrm{LS}} = \boldsymbol{z}$ 所得的 LS 解 $\hat{x}_{\mathrm{LS}}$ 与用 Householder QR 方法得到的解一样好. 也就是说, 形如 (5.3.4)~(5.3.6) 的结果是适用的, 参见 Björck and Paige(1992).

需要注意的是, 因为 MGS 总是对 $m$ 维向量进行运算, 而 Householder QR 处理更短的向量, 故前者比后者开销要稍大一些.

### 5.3.6 LS 问题的快速 Givens 解法

也可以应用快速 Givens 变换来解 LS 问题. 假定 $\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} = \boldsymbol{D}$ 是对角矩阵且

$$\boldsymbol{M}^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \boldsymbol{S}_1 \\ \mathbf{0} \end{bmatrix}\begin{matrix} n \\ m-n \end{matrix}$$

是上三角形矩阵. 如果

$$\boldsymbol{M}^{\mathrm{T}}\boldsymbol{b} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}\begin{matrix} n \\ m-n \end{matrix},$$

则对于任意 $\boldsymbol{x} \in \mathbb{R}^n$ 有

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = \|\boldsymbol{D}^{-1/2}\boldsymbol{M}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\|_2^2 = \left\|\boldsymbol{D}^{-1/2}\left(\begin{bmatrix} \boldsymbol{S}_1 \\ \mathbf{0} \end{bmatrix}\boldsymbol{x} - \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}\right)\right\|_2^2.$$

显然通过解非奇异的上三角方程组 $S_1x = c$ 可得到 $\boldsymbol{x}_{\mathrm{LS}}$.

可以证明, 用这种方式得到的计算解 $\hat{\boldsymbol{x}}_{\mathrm{LS}}$ 在 (5.3.4)~(5.3.6) 意义下是附近的 LS 问题的解. 这看上去很惊奇, 因为在计算过程中会出现大数. 每经过一次快速 Givens 变换, 加权矩阵 $\boldsymbol{D}$ 中的元素就可能翻倍. 然而, $\boldsymbol{D}$ 的增大正好可由 $\boldsymbol{M}$ 的增大来补偿, 因为在整个计算中 $\boldsymbol{D}^{-1/2}\boldsymbol{M}$ 总是正交的. 正是这一现象才使人们能得到很好的误差分析.

### 5.3.7 LS 问题的灵敏性

下面介绍扰动理论, 它有助于比较 LS 问题的法方程组法和 QR 分解法. 下面的定理考虑 LS 问题的解以及剩余是如何受 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的变化影响的. 为此, 要确定 LS 问题的条件数.

在分析过程中需用到两个显而易见的事实:

$$\|\boldsymbol{A}\|_2\|(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\|_2 = \kappa_2(\boldsymbol{A}),$$

$$\|\boldsymbol{A}\|_2^2\|(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\|_2 = \kappa_2(\boldsymbol{A})^2. \tag{5.3.7}$$

这两个方程可用 SVD 来验证.

**定理 5.3.1** 假定 $x, r, \hat{x}$ 和 $\hat{r}$ 满足

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 = \min, \quad \boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$$

$$\|(\boldsymbol{A} + \delta\boldsymbol{A})\hat{\boldsymbol{x}} - (\boldsymbol{b} + \delta\boldsymbol{b})\|_2 = \min, \quad \hat{\boldsymbol{r}} = (\boldsymbol{b} + \delta\boldsymbol{b}) - (\boldsymbol{A} + \delta\boldsymbol{A})\hat{\boldsymbol{x}}$$

其中 $\boldsymbol{A}$ 和 $\delta\boldsymbol{A}$ 属于 $\mathbb{R}^{m\times n}$ 且 $m \geqslant n, 0 \neq \boldsymbol{b}, \delta\boldsymbol{b}$ 属于 $\mathbb{R}^m$. 如果

$$\varepsilon = \max\left\{\frac{\|\delta\boldsymbol{A}\|_2}{\|\boldsymbol{A}\|_2}, \frac{\|\delta\boldsymbol{b}\|_2}{\|\boldsymbol{b}\|_2}\right\} < \frac{\sigma_n(\boldsymbol{A})}{\sigma_1(\boldsymbol{A})},$$

$$\sin(\theta) = \frac{\rho_{\mathrm{LS}}}{\|\boldsymbol{b}\|_2} \neq 1,$$

其中 $\rho_{\mathrm{LS}} = \|\boldsymbol{A}\boldsymbol{x}_{\mathrm{LS}} - \boldsymbol{b}\|_2$, 则

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leqslant \varepsilon\left\{\frac{2\kappa_2(\boldsymbol{A})}{\cos(\theta)} + \tan(\theta)\kappa_2(A)^2\right\} + O(\varepsilon^2) \tag{5.3.8}$$

$$\frac{\|\hat{\boldsymbol{r}} - \boldsymbol{r}\|_2}{\|\boldsymbol{b}\|_2} \leqslant \varepsilon(1 + 2\kappa_2(\boldsymbol{A}))\min(1, m-n) + O(\varepsilon^2). \tag{5.3.9}$$

**证明** 设 $\boldsymbol{E}$ 和 $f$ 定义为 $\boldsymbol{E} = \delta\boldsymbol{A}/\varepsilon$ 和 $f = \delta\boldsymbol{b}/\varepsilon$, 通过假设 $\|\delta\boldsymbol{A}\|_2 < \sigma_n(\boldsymbol{A})$ 和由定理 2.5.2, 对于所有 $t \in [0, \varepsilon]$, 有 $\mathrm{rank}(\boldsymbol{A} + t\boldsymbol{E}) = n$. 由此推出

$$(\boldsymbol{A} + t\boldsymbol{E})^{\mathrm{T}}(\boldsymbol{A} + t\boldsymbol{E})x(t) = (\boldsymbol{A} + t\boldsymbol{E})^{\mathrm{T}}(b + tf) \tag{5.3.10}$$

的解 $\boldsymbol{x}(t)$ 对所有 $t \in [0, \varepsilon]$ 是连续可微的. 由于 $\boldsymbol{x} = x(0)$ 和 $\hat{\boldsymbol{x}} = x(\varepsilon)$, 有

$$\hat{\boldsymbol{x}} = \boldsymbol{x} + \varepsilon\dot{\boldsymbol{x}}(0) + O(\varepsilon^2).$$

假设 $\boldsymbol{b} \neq \boldsymbol{0}$ 和 $\sin(\theta) \neq 1$ 保证了 $\boldsymbol{x}$ 是非零的, 故

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \varepsilon\frac{\|\dot{x}(0)\|_2}{\|\boldsymbol{x}\|_2} + O(\varepsilon^2). \tag{5.3.11}$$

为了对 $\|\dot{x}(0)\|_2$ 限界, 对 (5.3.10) 求导且令 $t = 0$, 得到

$$\boldsymbol{E}^{\mathrm{T}}\boldsymbol{A}x + \boldsymbol{A}^{\mathrm{T}}\boldsymbol{E}\boldsymbol{x} + \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\dot{x}(0) = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{f} + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{b},$$

即

$$\dot{x}(0) = (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{f} - \boldsymbol{E}\boldsymbol{x}) + (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{r}. \tag{5.3.12}$$

将此结果代入 (5.3.11), 取范数, 利用很易验证的不等式 $\|f\|_2 \leqslant \|b\|_2$ 和 $\|\boldsymbol{E}\|_2 \leqslant \|\boldsymbol{A}\|_2$, 我们得到

$$\frac{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leqslant \varepsilon\left\{\|\boldsymbol{A}\|_2\|(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\|_2\left(\frac{\|\boldsymbol{b}\|_2}{\|\boldsymbol{A}\|_2\|\boldsymbol{x}\|_2}+1\right)\right.$$
$$\left.+\frac{\rho_{\mathrm{LS}}}{\|\boldsymbol{A}\|_2\|\boldsymbol{x}\|_2}\|\boldsymbol{A}\|_2^2\|(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\|_2\right\}+O(\varepsilon^2).$$

由于 $\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})=0, \boldsymbol{A}\boldsymbol{x}$ 与 $\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}$ 是正交的, 故

$$\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2+\|\boldsymbol{A}\boldsymbol{x}\|_2^2=\|\boldsymbol{b}\|_2^2.$$

因此,

$$\|\boldsymbol{A}\|_2^2\|\boldsymbol{x}\|_2^2 \geqslant \|\boldsymbol{b}\|_2^2-\rho_{\mathrm{LS}}^2,$$

于是利用 (5.3.7) 有

$$\frac{\|\hat{\boldsymbol{x}}-\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leqslant \varepsilon\left\{\kappa_2(\boldsymbol{A})\left(\frac{1}{\cos(\theta)}+1\right)+\kappa_2(\boldsymbol{A})^2\frac{\sin(\theta)}{\cos(\theta)}\right\}+O(\varepsilon^2),$$

因而得到 (5.3.8).

为证 (5.3.9), 定义可微向量函数 $r(t)$ 为

$$r(t)=(\boldsymbol{b}+t\boldsymbol{f})-(\boldsymbol{A}+t\boldsymbol{E})x(t),$$

且观察到 $\boldsymbol{r}=r(0)$ 和 $\hat{\boldsymbol{r}}=r(\varepsilon)$. 利用 (5.3.12) 可以证明:

$$\dot{r}(0)=(\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}})(\boldsymbol{f}-\boldsymbol{E}\boldsymbol{x})-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{r}.$$

由于 $\|\hat{\boldsymbol{r}}-\boldsymbol{r}\|_2=\varepsilon\|\dot{r}(0)\|_2+O(\varepsilon^2)$, 我们有

$$\begin{aligned}\frac{\|\hat{\boldsymbol{r}}-\boldsymbol{r}\|_2}{\|\boldsymbol{b}\|_2} &= \frac{\varepsilon\|\dot{r}(0)\|_2}{\|\boldsymbol{b}\|_2}+O(\varepsilon^2)\\ &\leqslant \varepsilon\left\{\|\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\|_2\left(1+\frac{\|\boldsymbol{A}\|_2\|\boldsymbol{x}\|_2}{\|\boldsymbol{b}\|_2}\right)\right.\\ &\quad\left.+\|\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\|_2\|\boldsymbol{A}\|_2\frac{\rho_{\mathrm{LS}}}{\|\boldsymbol{b}\|_2}\right\}+O(\varepsilon^2).\end{aligned}$$

由

$$\|\boldsymbol{A}\|_2\|\boldsymbol{x}\|_2=\|\boldsymbol{A}\|_2\|\boldsymbol{A}^{+}\boldsymbol{b}\|_2 \leqslant \kappa_2(\boldsymbol{A})\|\boldsymbol{b}\|_2,$$
$$\rho_{\mathrm{LS}}=\|(\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}})\boldsymbol{b}\|_2 \leqslant \|\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\|_2\|\boldsymbol{b}\|_2,$$
$$\|(\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}})\|_2=\min(m-n,1),$$

故知不等式 (5.3.9) 成立. □

(5.3.8) 的上界的一个有趣特征是因子

$$\tan(\theta)\kappa_2(\boldsymbol{A})^2=\frac{\rho_{\mathrm{LS}}}{\sqrt{\|b\|_2^2-\rho_{\mathrm{LS}}^2}}\kappa_2(\boldsymbol{A})^2.$$

由此可以看出, 对非零剩余问题 $x_{\mathrm{LS}}$ 的敏感性将由条件数的平方来度量. 而余量的敏感性只是线性地依赖性于 $\kappa_2(A)$. 这种依赖性在例 5.3.1 中得到证实.

### 5.3.8 法方程组法与 QR 分解的比较

将解 LS 问题的法方程组法和 QR 分解法进行比较是有意义的. 回想我们的讨论中的主要观点:

- LS 问题解的灵敏性大致与 $\kappa_2(\boldsymbol{A})+\rho_{\mathrm{LS}}\kappa_2(\boldsymbol{A})^2$ 成正比.
- 法方程组法得到的解 $\hat{x}_{\mathrm{LS}}$ 的相对误差依赖于条件数的平方.
- QR 方法 (Householder, Givens, 细致的 MGS) 解一个附近的 LS 问题, 得到的解之相对误差约为 $\mathbf{u}(\kappa_2(\boldsymbol{A})+\rho_{\mathrm{LS}}\kappa_2(\boldsymbol{A})^2)$.

因此, 可得出结论: 如果 $\rho_{\mathrm{LS}}$ 很小且 $\kappa_2(A)$ 很大, 则法方程组法不会解一个附近问题, 它通常给出比稳定的 QR 方法精度低的 LS 解. 相反, 当用来解大余量和病态问题时, 用两种方法得到差不多的不精确的解.

最后, 我们给出关于 QR 法和法方程组法优劣的另外两个因素:

- 法方程组法在 $m \gg n$ 时只需进行一半的运算, 且不需要太大的存储空间.
- 由于应用于 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的 Cholesky 算法会在执行 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{R}$ 的向后消去过程 "之前" 中止, 因此 QR 方法适用于更大的矩阵类.

相信以上的讨论至少会使你认识到选择一个 "正确的" 算法是多么困难.

## 习 题

**5.3.1** 假定 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b},(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\boldsymbol{F})\hat{\boldsymbol{x}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b},2\|\boldsymbol{F}\|_2\leqslant\sigma_n(\boldsymbol{A})^2$. 证明如果 $\boldsymbol{r}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x},\hat{\boldsymbol{r}}=\boldsymbol{b}-\boldsymbol{A}\hat{\boldsymbol{x}}$, 则 $\hat{\boldsymbol{r}}-\boldsymbol{r}=\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\boldsymbol{F})^{-1}\boldsymbol{F}\boldsymbol{x}$ 和

$$\|\hat{\boldsymbol{r}}-\boldsymbol{r}\|_2\leqslant 2\kappa_2(\boldsymbol{A})\frac{\|\boldsymbol{F}_2\|}{\|\boldsymbol{A}\|_2}\|\boldsymbol{x}\|_2.$$

**5.3.2** 假定 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b},\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\hat{\boldsymbol{x}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}+\boldsymbol{f}$, 其中 $\|\boldsymbol{f}\|_2\leqslant c\mathbf{u}\|\boldsymbol{A}^{\mathrm{T}}\|_2\|\boldsymbol{b}\|_2$, 且 $\boldsymbol{A}$ 为列满秩的, 证明

$$\frac{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|_2}{\|\boldsymbol{x}\|_2}\leqslant c\mathbf{u}\kappa_2(\boldsymbol{A})^2\frac{\|\boldsymbol{A}^{\mathrm{T}}\|_2\|\boldsymbol{b}\|_2}{\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}\|}.$$

**5.3.3** 设 $\boldsymbol{A}\in\mathbb{R}^{m\times n},m>n,\boldsymbol{y}\in\mathbb{R}^m$, 定义 $\bar{\boldsymbol{A}}=[\boldsymbol{A}\ \boldsymbol{y}]\in\mathbb{R}^{m\times(n+1)}$, 证明 $\sigma_1(\bar{\boldsymbol{A}})\geqslant\sigma_1(\boldsymbol{A})$ 和 $\sigma_{n+1}(\bar{\boldsymbol{A}})\leqslant\sigma_n(\boldsymbol{A})$. 因此对矩阵增加一列后, 条件数增大.

**5.3.4** 令 $\boldsymbol{A}\in\mathbb{R}^{m\times n}(m\geqslant n),\boldsymbol{w}\in\mathbb{R}^n$, 定义

$$\boldsymbol{B}=\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{w}^{\mathrm{T}}\end{bmatrix},$$

证明 $\sigma_n(\boldsymbol{B})\geqslant\sigma_n(\boldsymbol{A}),\sigma_1(\boldsymbol{B})\leqslant\sqrt{\|\boldsymbol{A}\|_2^2+\|\boldsymbol{w}\|_2^2}$. 因此对矩阵增加一行后, 条件数可能变大或变小.

**5.3.5** (Cline 1973) 假定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 的秩为 $n$, 用列选主元的高斯消去法计算分解 $\boldsymbol{P}\boldsymbol{A}=\boldsymbol{L}\boldsymbol{U}$, 其中 $\boldsymbol{L}\in\mathbb{R}^{m\times n}$ 是单位下三角形矩阵, $\boldsymbol{U}\in\mathbb{R}^{n\times n}$ 是上三角形矩阵, $\boldsymbol{P}\in\mathbb{R}^{m\times m}$ 是置换矩阵. 指出如何利用习题 5.2.5 的分解来寻找一个向量 $\boldsymbol{z}\in\mathbb{R}^n$ 使得 $\|\boldsymbol{L}\boldsymbol{z}-\boldsymbol{P}\boldsymbol{b}\|_2$ 极小化. 证明如果 $\boldsymbol{U}\boldsymbol{x}=\boldsymbol{z}$, 则 $\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2$ 取极小值. 证明当 $m\leqslant 5n/3$ 时, 从浮点运算的角度在求解 LS 问题上此方法比 Householder QR 方法更有效.

**5.3.6** 在许多统计应用中都要用到矩阵 $C=(A^{\mathrm{T}}A)^{-1}$, 其中 $\mathrm{rank}(A)=n, C$ 称为方差–协方差矩阵. 假定 QR 分解 $A=QR$ 可行, 证明 (a)$C=(R^{\mathrm{T}}R)^{-1}$, (b) 给出一个算法在 $n^3/9$ 个 flop 内求解 $C$ 的对角线元, (c) 证明

$$R=\begin{bmatrix}\alpha & v^{\mathrm{T}}\\ 0 & S\end{bmatrix}\Rightarrow C=(R^{\mathrm{T}}R)^{-1}=\begin{bmatrix}(1+v^{\mathrm{T}}C_1v)/\alpha^2 & -v^{\mathrm{T}}C_1/\alpha\\ -C_1v/\alpha & C_1\end{bmatrix},$$

其中 $C_1=(S^{\mathrm{T}}S)^{-1}$.

(d) 利用 (c) 的结果, 给出一个算法用 $C$ 的上三角部分覆盖 $R$ 的上三角部分, 算法应需 $2n^3/3$ flops.

**5.3.7** 假定 $A\in\mathbb{R}^{n\times n}$ 是对称的, $r=b-Ax$, 其中 $r,b,x\in\mathbb{R}^n, x$ 非零. 指出如何计算出一个对称矩阵 $E\in\mathbb{R}^{n\times n}$, 使其 Frobenius 范数极小化且 $(A+E)x=b$. 提示, 利用 $[x\quad r]$ 的 QR 分解, 注意 $Ex=r\Rightarrow(Q^{\mathrm{T}}EQ)(Q^{\mathrm{T}}x)=Q^{\mathrm{T}}r$.

**5.3.8** 指出如何计算最接近 Toeplitsz 矩阵的循环矩阵. 用 Frobenius 范数定义距离.

## 本节注释与参考文献

我们只限于最小二乘近似并不是反对其他范数极小化. 有些时候应该对 $\|Ax-b\|_p$ 进行极小化, 其中 $p=1$ 和 $\infty$, 这方面的一些算法请参阅:

A. K. Cline(1976a). "A. Descent Method for the Uniform Solution to Overdetermined Systems of Equations," *SIAM J. Num. Anal.* 13, 293–309.

R. H. Bartels, A. R. Conn, and C. Charalambous(1978). "On Cline's Direct Method for Solving Overdetermined Linear Systems in the $L_\infty$ Sense," *SIAM J. Num. Anal.* 15, 255–270.

T. F. Coleman and Y. Li(1992). "A Globally and Quadratically Convergent Affine Scaling Method for Linear $L_1$ Problems," *Mathematical Programming*, 56, *Series A*, 189–222.

Y. Li(1993). "A Globally Convergent Method for $L_p$ Problems," *SIAM J. Optimization*, 3, 609–629.

Y. Zhang(1993). "A Primal-Dual Interior Point Approach for Computing the $L_1$ and $L_\infty$ Solutions of Overdetermined Linear Systems," *J. Optimization Theory and Applications*, 77, 323–341.

应用高斯变换求解 LS 问题因比用 Householder 变换和 Givens 变换代价小, 而吸引了很多注意力, 参阅:

G. Peters and J. H. Wilkinson(1970). "The Least Squares Problem and Pseudo-Inverses," *Comp. J.* 13, 309–316.

A. K. Cline(1973). "An Elimination Method for the Solution of Linear Least Squares Problems," *SIAM J. Num. Anal.* 10, 283–289.

R. J. Plemmons(1974). "Linear Least Squares by Elimination and MGS," *J. Assoc. Comp. Mach.* 21, 581–585.

LS 问题的许多重要分析和求解方法见:

G. H. Golub and J. H. Wilkinson(1966). "Note on the Iterative Refinement of Least Squares Solution," *Numer. Math.* 9, 139–248.

A. van der Sluis(1975). "Stability of the Solutions of Linear Least Squares Problem," *Numer. Math.* 23, 241–254.

Y. Saad(1986). "On the Condition Number of Some Gram Matrices Arising from Least Squares Approximation in the Complex Plane," *Numer. Math.* 48, 337–348.

Å. Björck (1987). "Stability Analysis of the Method of Seminormal Equations," *Lin. Alg. and Its Applic.* 88/89, 31–48.

J. Gluchowska and A. Smoktunowicz(1990). "Solving the Linear Least Squares Problem with Very High Relative Accuracy," *Computing* 45, 345–354.

Å. Björck (1991). "Component-wise Perturbation Analysis and Error Bounds for Linear Least Squares Solutions," *BIT* 31, 238–244.

Å. Björck and C. C. Paige (1992). "Loss and Recapture of Orthogonality in the Modified Gram-Schmidt Algorithm," *SIAM J. Matrix Anal. Appl.* 13, 176–190.

B. Waldén, R. Karlson, J. Sun(1995). "Optimal Backward Perturebation Bounds for the Linear Least Squares Problem," *Numerical Lin. Alg. with Applic.* 2, 271-286.

半正规的方程由 $R^TRx = A^Tb$(其中 $A = QR$) 给定. 在上述论文中, 通过求解半正规方程, 如果进行一次固定精度的迭代改善则可获得一个可接受的 LS 解.

用 MGS 方法求解 LS 问题的 Algol 算法见:

F. L. Bauer(1965), "Elimination with Weighted Row Combinations for Solving Linear Equations and Least Squares Problems," *Numer. Math.* 7, 338–352. See also Wilkinson and Reinsch (1971, 119–133.)

LS 问题经常具有特殊结构, 因此应当充分利用:

M. G. Cox(1981). "The Least Squares Solution of Overdetermined Linear Equations having Band or Augmented Band Structure," *IMA J. Num. Anal.* 1, 3–22.

G. Cybenko(1984). "The Numerical Stability of the Lattice Algorithm for Least Squares Linear Prediction Problems," *BIT* 24, 441–455.

P. C. Hansen and H. Gesmar(1993). "Fast Orthogonal Decomposition of Rank-Deficient Toeplitz Matrices," *Numerical Algorithms* 4, 151–166.

应用 Householder 矩阵求解稀疏 LS 问题时必须小心, 以防过量的填充:

J. K. Reid(1967). "A Note on the Least Squares Solution of a Band System of Linear Equations by Householder Reductions," *Comp. J.* 10, 188–189.

I. S. Duff and J. K. Reid(1976). "A Comparison of Some Methods for the Solution of Sparse Over-Determined Systems of Linear Equations," *J. Inst. Math. Applic.* 17, 267–280.

P. E. Gill and W. Murray(1976). "The Orthogonal Factorization of a Large Sparse Matrix," in *Sparse Matrix Computations*, ed. J. R. Bunch and D. J. Rose, Academic Press, New York, pp. 177–200.

L. Kaufman(1979). "Application of Dense Householder Transformations to a Sparse Matrix," *ACM Trans. Math. Soft.* 5, 442–451.

尽管应用了 Householder 反射后 QR 分解的计算更加高效了, 但有些场合用 Givens 旋转

更有效, 例如当 $\boldsymbol{A}$ 是稀疏矩阵时, 小心地应用 Givens 旋转可以把填充量降至最低.

I. S. Duff(1974). "Pivot Selection and Row Ordering in Givens Reduction on Sparse Matrices," *Computing* 13, 239–248.

J. A. George and M. T. Heath(1980). "Solution of Sparse Linear Least Squares Problems Using Givens Rotations," *Lin. Alg. and Its Applic.* 34, 69–83.

## 5.4 其他正交分解

如果 $\boldsymbol{A}$ 是秩亏损的, 则 QR 分解不一定能给出 $\text{ran}(\boldsymbol{A})$ 的一组正交基. 这个问题可通过计算经过列置换后的 $\boldsymbol{A}$ 之 QR 分解来解决, 即 $\boldsymbol{A\Pi} = \boldsymbol{QR}$, 其中 $\boldsymbol{\Pi}$ 是置换矩阵.

如果右乘一个一般正交矩阵 $\boldsymbol{Z}$, 则 $\boldsymbol{A}$ 中的"数据"可被进一步压缩

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{AZ} = \boldsymbol{T}.$$

存在 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 的有趣选取, 这些选取以及相应的列主元的 QR 分解正是本节要讨论的.

### 5.4.1 秩亏损: 列选主的 QR 分解

如果 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 且 $\text{rank}(\boldsymbol{A}) < n$, 则 QR 分解不一定产生 $\text{ran}(\boldsymbol{A})$ 的一组正交基. 例如, 如果 $\boldsymbol{A}$ 有三列且

$$\boldsymbol{A} = [a_1, a_2, a_3] = [q_1, q_2, q_3]\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

是它的 QR 分解, 则 $\text{rank}(\boldsymbol{A}) = 2$, 但 $\text{ran}(\boldsymbol{A})$ 和三个子空间 $\text{span}\{q_1, q_2\}, \text{span}\{q_1, q_3\}$ 和 $\text{span}\{q_2, q_3\}$ 中任何一个都不相等.

幸运的是, 可对 Householder QR 分解算法 (算法 5.2.1) 做简单修改来生成 $\text{ran}(\boldsymbol{A})$ 的正交基. 修改的算法计算分解

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A\Pi} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ 0 & 0 \end{bmatrix}\begin{matrix} r \\ m-r \end{matrix} \qquad (5.4.1)$$
$$\qquad\qquad\qquad r \qquad n-r$$

其中 $r = \text{rank}(\boldsymbol{A}), \boldsymbol{Q}$ 是正交的, $\boldsymbol{R}_{11}$ 是上三角形矩阵且是非奇异的, $\boldsymbol{\Pi}$ 是置换矩阵. 如果按列划分 $\boldsymbol{A\Pi} = [a_{c_1}, \cdots, a_{c_n}]$ 和 $\boldsymbol{Q} = [q_1, \cdots, q_m]$, 则对 $k = 1:n$ 有

$$a_{c_k} = \sum_{i=1}^{\min\{r,k\}} r_{ik}q_i \in \text{span}\{q_1, \cdots, q_r\},$$

这意味

$$\text{ran}(A) = \text{span}\{q_1, \cdots, q_r\}.$$

矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{\Pi}$ 分别是 Householder 矩阵乘积和初等置换矩阵的乘积. 假定对某个 $k$ 我们已计算了 Householder 矩阵 $\boldsymbol{H}_1\cdots\boldsymbol{H}_{k-1}$ 和置换矩阵 $\boldsymbol{\Pi}_1\cdots\boldsymbol{\Pi}_{k-1}$ 使得

$$(\boldsymbol{H}_{k-1}\cdots\boldsymbol{H}_1)\boldsymbol{A}(\boldsymbol{\Pi}_1\cdots\boldsymbol{\Pi}_{k-1}) =$$

$$\boldsymbol{R}^{(k-1)} = \begin{bmatrix} \boldsymbol{R}_{11}^{(k-1)} & \boldsymbol{R}_{12}^{(k-1)} \\ 0 & \boldsymbol{R}_{22}^{(k-1)} \end{bmatrix} \begin{matrix} k-1 \\ m-k+1 \end{matrix}, \quad (5.4.2)$$
$$\qquad\quad k-1 \quad n-k+1$$

其中 $\boldsymbol{R}_{11}^{(k-1)}$ 是非奇异上三角形矩阵. 现在假定

$$\boldsymbol{R}_{22}^{(k-1)} = \left[z_k^{(k-1)}, \cdots, z_n^{(k-1)}\right]$$

是按列划分的, 且令 $p \geqslant k$ 是使得

$$\|z_p^{(k-1)}\|_2 = \max\{\|z_k^{(k-1)}\|_2, \cdots, \|z_n^{(k-1)}\|_2\} \quad (5.4.3)$$

的最小下标. 注意, 如果 $k-1=\text{rank}(\boldsymbol{A})$, 则这个极大值是零, 从而计算至此结束. 否则, 令 $\boldsymbol{\Pi}_k$ 是互换第 $p$ 列和第 $k$ 列的 $n\times n$ 单位矩阵, 并且确定 Householder 矩阵 $\boldsymbol{H}_k$ 使得对 $\boldsymbol{R}^{(k)} = \boldsymbol{H}\boldsymbol{R}^{(k-1)}\boldsymbol{\Pi}_k$ 有 $R^{(k)}(k+1:m,k)=0$. 换句话说, $\boldsymbol{\Pi}_k$ 把 $R_{22}^{(k-1)}$ 的最大列移到前面, $\boldsymbol{H}_k$ 把它的非对角元化为零.

如果利用对于任何正交矩阵 $\boldsymbol{Q}\in\mathbb{R}^{s\times s}$ 均成立的性质

$$\boldsymbol{Q}^{\mathrm{T}}z = \begin{bmatrix} \alpha \\ \omega \end{bmatrix} \begin{matrix} 1 \\ s-1 \end{matrix} \Rightarrow \|\omega\|_2^2 = \|z\|_2^2 - \alpha^2,$$

那么不必每一步重新计算列范数. 因为我们可以通过修正旧的列范数来得到新的列范数, 即

$$\|z^{(j)}\|_2^2 = \|z^{(j-1)}\|_2^2 - r_{kj}^2,$$

这使列选主的工作量由 $O(mn^2)$ 个 flop 减少到 $O(mn)$ 个 flop. 综上所述, 我们得到由 Businger and Golub(1965) 给出的算法.

**算法 5.4.1 (列选主的 Householder QR 分解)**　给定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 且 $m\geqslant n$, 本算法计算 $r=\text{rank}(\boldsymbol{A})$ 和分解 (5.4.1), 其中 $\boldsymbol{Q}=\boldsymbol{H}_1\cdots\boldsymbol{H}_r, \boldsymbol{\Pi}=\boldsymbol{\Pi}_1\cdots\boldsymbol{\Pi}_r$.$\boldsymbol{A}$ 的上三角部分被 $\boldsymbol{R}$ 的上三角部分覆盖, 第 $j$ 个 Householder 向量的 $j+1:m$ 分量存储于 $A(j+1:m,j)$ 中, 置换矩阵 $\boldsymbol{\Pi}$ 由整数向量 piv 来标记. 具体地说, $\boldsymbol{\Pi}_j$ 是将第 $j$ 行和第 piv$(j)$ 行互换的单位矩阵.

```
for j = 1 : n
    c(j) = A(1 : m, j)^T A(1 : m, j)
end
r = 0; τ = max{c(1), ··· , c(n)}
求最小 k(1 ≤ k ≤ n) 使得 c(k) = τ
while τ > 0
    r = r + 1
    piv (r) = k; A(1 : m, r) ↔ A(1 : m, k); c(r) ↔ c(k)
    [v, β] = house(A(r : m, r))
    A(r : m, r : n) = (I_{m-r+1} − βvv^T)A(r : m, r : n)
```

```
        A(r+1:m,r) = v(2:m-r+1)
        for i = r+1:n
            c(i) = c(i) - A(r,i)^2
        end
        if r < n
            τ = max{c(r+1),···,c(n)}
            求最小 k(r+1 ⩽ k ⩽ n) 使得 c(k) = τ.
        else
            τ = 0
        end
    end
```

本算法需要 $4mnr-2r^2(m+n)+4r^3/3$ 个 flop, 其中 $r=\text{rank}(\boldsymbol{A})$. 如同非选主的算法 5.2.1 一样, 正交矩阵 $\boldsymbol{Q}$ 以分解的形式存储在 $\boldsymbol{A}$ 的次对角线部分.

**例 5.4.1** 如果用算法 5.4.1 来解

$$\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\ 1 & 5 & 6\\ 1 & 8 & 9\\ 1 & 11 & 12\end{bmatrix},$$

则 $\boldsymbol{\Pi}=[e_3 \quad e_2 \quad e_1]$, 保留 3 位有效数我们有

$$\boldsymbol{A\Pi}=\boldsymbol{QR}=\begin{bmatrix}-0.182 & -0.816 & 0.514 & 0.191\\ -0.365 & 0.408 & -0.827 & 0.129\\ 0.548 & 0.000 & 0.113 & -0.829\\ -0.730 & 0.408 & 0.200 & 0.510\end{bmatrix}$$
$$\times\begin{bmatrix}-16.4 & -14.600 & -1.820\\ 0.0 & 0.816 & -0.816\\ 0.0 & 0.000 & 0.000\\ 0.0 & 0.000 & 0.000\end{bmatrix}.$$

### 5.4.2 完全正交分解

由算法 5.4.1 产生的矩阵 $\boldsymbol{R}$, 如果从右边用一组适当的 Householder 矩阵相乘则可进一步约化. 具体地说, 我们可用算法 5.2.1 来计算

$$\boldsymbol{Z}_r\cdots\boldsymbol{Z}_1\begin{bmatrix}\boldsymbol{R}_{11}^{\mathrm{T}}\\ \boldsymbol{R}_{12}^{\mathrm{T}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{T}_{11}^{\mathrm{T}}\\ \boldsymbol{0}\end{bmatrix}\begin{matrix}r\\ n-r\end{matrix} \tag{5.4.4}$$

其中 $\boldsymbol{Z}_i$ 是 Householder 变换, $\boldsymbol{T}_{11}^{\mathrm{T}}$ 是上三角形矩阵. 于是有

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{AZ}=\boldsymbol{T}=\begin{bmatrix}\boldsymbol{T}_{11} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{matrix}r\\ m-r\end{matrix} \tag{5.4.5}$$
$$\qquad\qquad r \quad n-r$$

其中 $\boldsymbol{Z}=\boldsymbol{\Pi Z}_1\cdots\boldsymbol{Z}_r$. 我们称任何这种形式的分解为**完全正交分解**. 注意到 $\text{null}(A)=\text{ran}(Z(1:n,r+1:n))$, 有关利用 (5.4.4) 的结构之细节请参见习题 5.2.5.

### 5.4.3 双对角化

假定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 且 $m \geqslant n$. 下面我们给出如何计算正交矩阵 $\boldsymbol{U}_B$ $(m\times m)$ 和 $\boldsymbol{V}_B$ $(n\times n)$ 使得

$$\boldsymbol{U}_B^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_B = \left[\begin{array}{ccccc} d_1 & f_1 & 0 & \cdots & 0 \\ 0 & d_2 & f_2 & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & & d_{n-1} & f_{n-1} \\ 0 & \cdots & & 0 & d_n \\ \hline & & \mathbf{0} & & \end{array}\right]. \tag{5.4.6}$$

$\boldsymbol{U}_B = \boldsymbol{U}_1\cdots\boldsymbol{U}_n$ 和 $\boldsymbol{V}_B = \boldsymbol{V}_1\cdots\boldsymbol{V}_{n-2}$ 都可由 Householder 矩阵的乘积给出:

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{U}_1} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{V}_1}$$

$$\begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{U}_2} \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{V}_2}$$

$$\begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{U}_3} \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \xrightarrow{\boldsymbol{U}_4} \begin{bmatrix} \times & \times & 0 & 0 \\ 0 & \times & \times & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

一般地, $\boldsymbol{U}_k$ 在第 $k$ 列引入零元, 而 $\boldsymbol{V}_k$ 在第 $k$ 行引进零元. 整个算法如下.

**算法 5.4.2 (Householder 双对角化)** 给定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 且 $m \geqslant n$, 本算法以 $\boldsymbol{U}_B^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_B = \boldsymbol{B}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{B}$ 是上两对角矩阵, 且 $\boldsymbol{U}_B = \boldsymbol{U}_1\cdots\boldsymbol{U}_n, \boldsymbol{V}_B = \boldsymbol{V}_1\cdots\boldsymbol{V}_{n-2}, \boldsymbol{U}_j$ 的 Householder 向量的基本部分存储于 $A(j+1:m,j)$ 中, $\boldsymbol{V}_j$ 的 Householder 向量的基本部分存储于 $A(j,j+2:n)$ 中.

**for**$=j=1:n$

  $[v,\beta] =$ **house**$(A(j:m,j))$

  $A(j:m,j:n) = (I_{m-r+1} - \beta vv^{\mathrm{T}})A(j:m,j:n)$

  $A(j+1:m,j) = v(2:m-j+1)$

  **if** $j \leqslant n-2$

$[v, \beta] = \textbf{house}(A(j, j+1:n)^{\mathrm{T}})$
$A(j:m, j+1:n) = A(j:m, j+1:n)(I_{n-j} - \beta vv^{\mathrm{T}})$
$A(j, j+2:n) = v(2:n-j)^{\mathrm{T}}$
**end**
**end**

本算法需要 $4mn^2 - 4n^3/3$ 个 flop. Golub and Kahan(1965) 使用了此技术, 并首次提出了双对角化的概念. 如果需要 $\boldsymbol{U}_B$ 和 $\boldsymbol{V}_B$ 的显式表达式, 二者可分别在 $4m^2n - 4n^3/n$ 个 flop 和 $4n^3/3$ 个 flop 内累积得到. $\boldsymbol{A}$ 的双对角化与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的三对角化密切相关, 参见 8.2.1 节.

**例 5.4.2** 如果用算法 5.4.2 来解

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix},$$

则用 3 位有效数字计算有

$$\hat{\boldsymbol{B}} = \begin{bmatrix} 12.8 & 21.8 & 0 \\ 0 & 2.24 & -0.613 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \hat{\boldsymbol{V}}_B = \begin{bmatrix} 1.00 & 0.00 & 0.00 \\ 0.00 & -0.667 & -0.745 \\ 0.00 & -0.745 & 0.667 \end{bmatrix},$$

$$\hat{\boldsymbol{U}}_B = \begin{bmatrix} -0.0776 & -0.833 & 0.392 & -0.383 \\ -0.311 & -0.451 & -0.238 & 0.802 \\ -0.543 & -0.069 & 0.701 & -0.457 \\ -0.776 & 0.312 & 0.547 & 0.037 \end{bmatrix}.$$

### 5.4.4 R 双对角化

当 $m \gg n$ 时, 如在使用算法 5.4.2 之前首先对 $\boldsymbol{A}$ 进行上三对角化, 则会得到一个更加快速的双对角化算法. 具体地说, 假定我们计算一个正交矩阵 $\boldsymbol{Q} \in \mathbb{R}^{m\times m}$ 使得

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{0} \end{bmatrix}$$

是上三角形矩阵. 然后对方阵 $\boldsymbol{R}_1$ 进行双对角化

$$\boldsymbol{U}_R^{\mathrm{T}}\boldsymbol{R}_1\boldsymbol{V}_B = \boldsymbol{B}_1.$$

这里 $\boldsymbol{U}_R$ 和 $\boldsymbol{V}_B$ 是 $n \times n$ 正交矩阵, $\boldsymbol{B}_1$ 是 $n \times n$ 上双对角矩阵. 如果 $\boldsymbol{U}_B = \boldsymbol{Q}\,\mathrm{diag}(U_R, \boldsymbol{I}_{m-n})$, 则

$$U^{\mathrm{T}}AV = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \equiv B$$

是 $A$ 的双对角化.

以这种方式计算双对角化的思想是 Lawson and Hansen(1974, 119 页) 提出的, Chan(1982) 对此做了更全面的分析. 我们称此方法为 R 双对角化, 将它所需的 flop 数 $(2mn^2 + 2n^3)$ 与算法 5.4.2 所需的 flop 数 $(4mn^2 - 4n^3/3)$ 相比, 可看出在 $m \geqslant 5n/3$ 时, 它所需的计算量要少一些.

### 5.4.5 SVD 及其计算

一旦完成了对 $A$ 的双对角化, Golub-Reinsch 的 SVD 算法的下一步是对 $B$ 的超对角线元素进行消零. 这是一个迭代过程, 可由 Golub and Kahan(1965) 的算法来完成. 但由于对此迭代法的讨论需有关对称特征值的知识, 我们将其放在 8.6 节中讲述. 现在只指出它能够计算满足

$$U_{\Sigma}^{\mathrm{T}}BV_{\Sigma} = \Sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_n) \in \mathbb{R}^{m\times n}$$

的正交矩阵 $U_\Sigma$ 和 $V_\Sigma$. 通过定义 $U = U_B U_\Sigma$ 和 $V = V_B V_\Sigma$, 可以看出 $U^{\mathrm{T}}AV = \Sigma$ 是 $A$ 的 SVD. 算法中这部分计算的 flop 数与 SVD 分解到什么程度有关. 例如, 求解 LS 问题时, $U^{\mathrm{T}}$ 从来不用显式求出, 只是随着 $U^{\mathrm{T}}$ 的产生, 将它作用到 $b$ 上即可. 在其他应用中, 只需要矩阵 $U_1 = U(:, 1:n)$. 共计有六种可能情况, 每种情况下 SVD 算法所需的工作量总结在表 5.4.1 中. 由于存在两种双对角化的方法, 因此有两栏的 flop 数. 如果通过算法 5.4.2 实现双对角化, 则产生 Golub-Reinsch(1970)SVD 算法. 而如采用 R 双对角化, 我们得到 Chan(1982a) 详细给出的 R-SVD 算法. 通过比较表中的值 (只是工作量的近似估计), 我们的结论是: 除非 $m \approx n$, R-SVD 方法更有效.

**表 5.4.1**

| 需　要 | Golub-Reinsch SVD | R-SVD |
|---|---|---|
| $\Sigma$ | $4mn^2 - 4n^3/3$ | $2mn^2 + 2n^3$ |
| $\Sigma, V$ | $4mn^2 + 8n^3$ | $2mn^2 + 11n^3$ |
| $\Sigma, U$ | $4m^2n - 8mn^2$ | $4m^2n + 13n^3$ |
| $\Sigma, U_1$ | $14mn^2 - 2n^3$ | $6mn^2 + 11n^3$ |
| $\Sigma, U, V$ | $4m^2n + 8mn^2 + 9n^3$ | $4m^2n + 22n^3$ |
| $\Sigma, U_1, V$ | $14mn^2 + 8n^3$ | $6mn^2 + 20n^3$ |

## 习　题

**5.4.1** 假设 $A \in \mathbb{R}^{m\times n}$ 且 $m < n$. 给出一个算法计算分解

$$U^{\mathrm{T}}AV = [B \quad O],$$

其中 $B$ 是 $m \times m$ 的上双对角矩阵. (提示: 利用 Householder 矩阵获得

$$\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & \times & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \end{bmatrix}$$

的形式, 然后自右开始进行 Givens 旋转来向上"追赶"第 $(m+1)$ 列的 $(m, m+1)$ 元素.)

**5.4.2** 指出如何用 Givens 旋转对一个 $n \times n$ 上三角形矩阵进行双对角化.

**5.4.3** 指出如何用 Givens 旋转对三对角矩阵 $\boldsymbol{T} \in \mathbb{R}^{n \times n}$ 进行双对角化.

**5.4.4** 令 $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ 且假定 $\boldsymbol{v} \neq 0$ 满足 $\|\boldsymbol{A}v\|_2 = \sigma_n(\boldsymbol{A})\|\boldsymbol{v}\|_2$. 设 $\boldsymbol{\Pi}$ 是满足当 $\boldsymbol{\Pi}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{\omega}$ 时有 $|\omega_n| = \|\omega\|_\infty$ 的置换矩阵. 证明如果 $\boldsymbol{A\Pi} = \boldsymbol{QR}$ 是 $\boldsymbol{A\Pi}$ 的 QR 分解, 则 $|r_{nn}| \leqslant \sqrt{n}\sigma_n(\boldsymbol{A})$. 因此总是存在置换矩阵 $\boldsymbol{\Pi}$ 使得 $\boldsymbol{A\Pi}$ 的 QR 分解接近秩亏损.

**5.4.5** 给定 $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^m, \boldsymbol{Q} \in \mathbb{R}^{m \times m}, \boldsymbol{Q}$ 是正交矩阵, 证明如果

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x} = \begin{bmatrix} \alpha \\ u \end{bmatrix} \begin{matrix} 1 \\ m-1 \end{matrix}, \quad \boldsymbol{Q}^{\mathrm{T}}y = \begin{bmatrix} \beta \\ v \end{bmatrix} \begin{matrix} 1 \\ m-1 \end{matrix},$$

则 $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} - \alpha\beta$.

**5.4.6** 给定 $\boldsymbol{A} = [a_1, \cdots, a_n] \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$, 对 $\boldsymbol{A}$ 的列向量的任意子集 $\{a_{c1}, \cdots, a_{ck}\}$, 定义

$$\operatorname{res}[a_{c1}, \cdots, a_{ck}] = \min_{x \in \mathbb{R}^k} \|[a_{c1}, \cdots, a_{ck}]x - b\|_2$$

描述一种不同于算法 5.4.1 的选主元方法使得: 如果最终分解有 QR=$\boldsymbol{A\Pi} = [a_{c1}, \cdots, a_{ck}]$, 则对于 $k = 1:n$ 有

$$\operatorname{res}[a_{c1}, \cdots, a_{ck}] = \min_{i \geqslant k} \quad \operatorname{res}[a_{c1}, \cdots, a_{ck-1}, a_{ci}]$$

## 本节注释与参考文献

完全正交分解的讨论见:

R. J. Hanson and C. L. Lawson(1969), "Extensions and Applications of the Householder Algorithm for Solving Linear Least Square Problems," *Math. Comp.* 23, 787–812.

P. A. Wedin(1973). "On the Almost Rank-Deficient Case of the Least Squares Problem," *BIT* 13, 344–354.

G. H. Golub and V. Pereyra(1976). "Differentiation of Pseudo-Inverses, Separable Nonlinear Least Squares Problems and Other Tales," in *Generalized Inverses and Applications*, ed. M. Z. Nashed, Academic Press, New York, pp. 303–324.

SVD 的计算在 8.6 节详细介绍, 但以下是有关的标准参考文献:

G. H. Golub and W. Kahan(1965). "Calculating the Singular Values and Pseudo-Inverse of a Matrix," *SIAM J. Num. Anal.* 2, 205–224.

P. A. Businger and G. H. Golub(1969). "Algorithm 358: Singular Value Decomposition of the Complex Matrix," *Comm. ACM* 12, 564–565.

G. H. Golub and C. Reinsch(1970). "Singular Value Decomposition and Least Squares Solutions," *Numer. Math.* 14, 403–420. See also Wilkinson and Reinsch (1971, pp. 1334– 1351).

T. F. Chan(1982). "An Improved Algorihtm for Computing the Singular Value Decomposition," *ACM Trans. Math. Soft.* 8, 72–83.

列选主的 QR 分解最早在下面文章中讨论:

P. A. Businger and G. H. Golub(1965). "Linear Least Squares Solutions by Householder Transformations," *Numer. Math.* 7, 269–276. See also Wilkinson and Reinsch(1971, pp. 11–18).

很难决定算法何时停止. 在秩亏损的问题中, 获得有关 $\boldsymbol{R}$ 的上三角形矩阵的最小奇异值的信息将很有帮助, 这可通过用 3.5.4 节的技巧或下述文章中的方法来实现:

I. Karasalo(1974). "A Criterion for Truncation of the QR Decomposition Algorithm for the Singular Linear Least Squares Problem," *BIT* 14, 156–166.

N. Anderson and I. Karasalo(1975). "On Computing Bounds for the Least Singular Value of a Triangular Matrix," *BIT* 15, 1–4.

用 QR 分解来估计秩的其他问题可见文献:

L. V. Foster(1986). "Rank and Null Space Calculations Using Matrix Decomposition without Column Interchanges," *Lin. Alg. and Its Aplic.* 74. 47–71.

T. F. Chan(1987). "Rank Revealing QR Factorizations," *Lin. Alg. and Its Applic.* 88/89, 67–82.

T. F. Chan and P. Hansen(1992). "Some Applications of the Rank Revealing QR Factorzation," *SIAM J. Sci, and Stat. Comp.* 13, 727–741.

J. L. Barlow and U. B. Vemulapati(1992). "Rank Detection Methods for Sparse Matrices," *SIAM J. Matrix. Anal. Appl.* 13. 1279–1297.

T-M. Hwang, W-W. Lin, and E. K. Yang(1992). "Rank-Revealing LU Factorizations," *Lin, Alg. and Its Applic.* 175. 115–141.

C. H. Bischof and P. C. Hansen(1992). "A Block Algorithm for Computing Rank-Revealing QR Factorizations," *Numerical Algorithms* 2, 371–392.

S. Chandrasekaren and I. C. F. Ipsen(1994). "On Rank-Revealing Factorizations," *SIAM J. Matrix Anal. Appl.* 15, 592–622.

R. D. Fierro and P. C. Hansen(1995). "Accuracy of TSVD Solutions Computed from Rank-Revealing Decompositions," *Numer. Math.* 70, 453–472.

## 5.5 秩亏损的 LS 问题

如果 $\boldsymbol{A}$ 是秩亏损的, 则 LS 问题有无穷多个解, 我们必须采用特殊的技术. 这些技术必须考虑确定数值秩这一难题.

给出一些 SVD 的预备知识后, 我们说明列选主的 QR 方法可用来确定使得 $\boldsymbol{A}x_B$ 是 $r = \text{rank}(\boldsymbol{A})$ 列的线性组合的极小点 $x_B$. 然后讨论可由 SVD 得出的极小 2 范数解.

### 5.5.1 极小范数解

假定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 且 $\text{rank}(\boldsymbol{A}) = r < n$. 秩亏损 LS 问题有无穷多个解, 这是因为如果 $\boldsymbol{x}$ 是一个极小解, $\boldsymbol{z} \in \text{null}(\boldsymbol{A})$, 则 $\boldsymbol{x}+\boldsymbol{z}$ 也是一个极小解. 所有极小解的集合

$$\chi = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2 = \min\}$$

是凸的, 因为如 $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \chi$ 且 $\lambda \in [0,1]$, 则

$$\begin{aligned}\|\boldsymbol{A}(\lambda\boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) - \boldsymbol{b}\|_2 &\leqslant \lambda\|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{b}\|_2 + (1-\lambda)\|\boldsymbol{A}\boldsymbol{x}_2 - \boldsymbol{b}\|_2 \\ &= \min\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2.\end{aligned}$$

因此, $\lambda\boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2 \in \chi$. 所以, $\chi$ 中有唯一元素具有极小 2 范数, 用 $x_{\text{LS}}$ 表示这个解 (注意在满秩情况时, 只有一个 LS 解, 故它必须有极小 2 范数. 因此, 这与 5.3 节的记号一致).

### 5.5.2 完全正交分解和 $\boldsymbol{x}_{\text{LS}}$

任何完全正交分解都可用来计算 $x_{\text{LS}}$. 具体地说, 如果 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 是正交矩阵且满足

$$\boldsymbol{Q}^{\text{T}}\boldsymbol{A}\boldsymbol{Z} = \boldsymbol{T} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}\begin{matrix} r \\ m-r \end{matrix}, \quad r = \text{rank}(\boldsymbol{A}),$$
$$\qquad\qquad\qquad\quad r \quad n-r$$

则

$$\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = \|(\boldsymbol{Q}^{\text{T}}\boldsymbol{A}\boldsymbol{Z})\boldsymbol{Z}^{\text{T}}\boldsymbol{x} - \boldsymbol{Q}^{\text{T}}\boldsymbol{b}\|_2^2 = \|\boldsymbol{T}_{11}\boldsymbol{\omega} - \boldsymbol{c}\|_2^2 + \|\boldsymbol{d}\|_2^2,$$

其中

$$\boldsymbol{Z}^{\text{T}}\boldsymbol{x} = \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{y} \end{bmatrix}\begin{matrix} r \\ n-r \end{matrix}, \quad \boldsymbol{Q}^{\text{T}}\boldsymbol{b} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{d} \end{bmatrix}\begin{matrix} r \\ m-r \end{matrix}.$$

很显然, 如果 $\boldsymbol{x}$ 使平方和极小, 那么必定有 $\boldsymbol{w} = \boldsymbol{T}^{-1}\boldsymbol{c}$. 因为要使 $\boldsymbol{x}$ 的 2 范数极小, $y$ 必须是零, 并且有

$$\boldsymbol{x}_{\text{LS}} = \boldsymbol{Z}\begin{bmatrix} \boldsymbol{T}_{11}^{-1}\boldsymbol{c} \\ \boldsymbol{0} \end{bmatrix}.$$

### 5.5.3 SVD 和 LS 问题

当然, SVD 是非常有启示性的完全正交分解. 它提供了 $\boldsymbol{x}_{\text{LS}}$ 的一个简洁表达式和极小剩余量的范数 $\rho_{\text{LS}} = \|\boldsymbol{A}\boldsymbol{x}_{\text{LS}} - \boldsymbol{b}\|_2$.

**定理 5.5.1** 假定 $\boldsymbol{U}^{\text{T}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{\Sigma}$ 是 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 的 SVD 且 $r = \text{rank}(\boldsymbol{A})$. 如果 $\boldsymbol{U} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m]$ 和 $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n]$ 是按列划分的, $\boldsymbol{b} \in \mathbb{R}^m$, 则

$$\boldsymbol{x}_{\text{LS}} = \sum_{i=1}^{r} \frac{\boldsymbol{u}_i^{\text{T}}\boldsymbol{b}}{\sigma_i}\boldsymbol{v}_i \tag{5.5.1}$$

使 $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$ 极小化, 且是所有极小点中 2 范数最小的. 而且

$$\rho_{\text{LS}}^2 = \|\boldsymbol{A}\boldsymbol{x}_{\text{LS}} - \boldsymbol{b}\|_2^2 = \sum_{i=r+1}^{m} (\boldsymbol{u}_i^{\text{T}}\boldsymbol{b})^2. \tag{5.5.2}$$

**证明** 对于任意 $\boldsymbol{x}\in\mathbb{R}^n$, 我们有

$$\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2=\|(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V})(\boldsymbol{V}^{\mathrm{T}}\boldsymbol{x})-\boldsymbol{U}^{\mathrm{T}}\boldsymbol{b}\|_2^2=\|\boldsymbol{\Sigma}\boldsymbol{\alpha}-\boldsymbol{U}^{\mathrm{T}}\boldsymbol{b}\|_2^2$$

$$=\sum_{i=1}^{r}(\sigma_i\alpha_i-\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b})^2+\sum_{i=r+1}^{m}(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b})^2,$$

其中 $\boldsymbol{\alpha}=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{x}$. 很显然, 若 $\boldsymbol{x}$ 是 LS 问题的解, 则对 $i=1:r$ 有 $\alpha_i=(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}/\sigma_i)$. 若令 $\alpha(r+1:n)=0$, 则得到的 $\boldsymbol{x}$ 显然具有极小 2 范数. □

### 5.5.4 广义逆

如果我们定义矩阵 $\boldsymbol{A}^+\in\mathbb{R}^{n\times m}$ 为 $\boldsymbol{A}^+=\boldsymbol{V}\boldsymbol{\Sigma}^+\boldsymbol{U}^{\mathrm{T}}$, 其中

$$\boldsymbol{\Sigma}^+=\mathrm{diag}\left(\frac{1}{\sigma_1},\cdots,\frac{1}{\sigma_r},0,\cdots,0\right)\in\mathbb{R}^{n\times m},\quad r=\mathrm{rank}(\boldsymbol{A}),$$

那么 $\boldsymbol{x}_{\mathrm{LS}}=\boldsymbol{A}^+\boldsymbol{b}$ 且 $\rho_{\mathrm{LS}}=\|(\boldsymbol{I}-\boldsymbol{A}\boldsymbol{A}^+)\boldsymbol{b}\|_2$. $\boldsymbol{A}^+$ 称为 $\boldsymbol{A}$ 的广义逆(Pseudo-inverse). 它是

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times m}}\|\boldsymbol{A}\boldsymbol{X}-\boldsymbol{I}_m\|_F \tag{5.5.3}$$

的唯一的极小 Frobenius 范数解. 如果 $\mathrm{rank}(\boldsymbol{A})=n$, 则 $\boldsymbol{A}^+=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}$, 而如果 $m=n=\mathrm{rank}(\boldsymbol{A})$, 则 $\boldsymbol{A}^+=\boldsymbol{A}^{-1}$. 通常, $\boldsymbol{A}^+$ 定义为满足如下四个 Moore-Penrose 条件的唯一矩阵 $\boldsymbol{X}\in\mathbb{R}^{n\times m}$:

(i) $\boldsymbol{A}\boldsymbol{X}\boldsymbol{A}=\boldsymbol{A}$; (ii) $\boldsymbol{X}\boldsymbol{A}\boldsymbol{X}=\boldsymbol{X}$;

(iii) $(\boldsymbol{A}\boldsymbol{X})^{\mathrm{T}}=\boldsymbol{A}\boldsymbol{X}$; (iv) $(\boldsymbol{X}\boldsymbol{A})^{\mathrm{T}}=\boldsymbol{X}\boldsymbol{A}$.

这些条件相当于要求 $\boldsymbol{A}\boldsymbol{A}^+$ 和 $\boldsymbol{A}^+\boldsymbol{A}$ 分别是 $\mathrm{ran}(\boldsymbol{A})$ 和 $\mathrm{ran}(\boldsymbol{A}^{\mathrm{T}})$ 上的正交投影. 事实上, $\boldsymbol{A}\boldsymbol{A}^+=\boldsymbol{U}_1\boldsymbol{U}_1^{\mathrm{T}}$, 其中 $\boldsymbol{U}_1=U(1:m,1:r)$, 和 $\boldsymbol{A}^+\boldsymbol{A}=\boldsymbol{V}_1\boldsymbol{V}_1^{\mathrm{T}}$, 其中 $\boldsymbol{V}_1=V(1:n,1:r)$.

### 5.5.5 一些敏感性问题

在 5.3 节我们分析了满秩 LS 问题的敏感性. 定理 5.3.1 对这种情形下 $\boldsymbol{x}_{\mathrm{LS}}$ 的性质做了总结. 如果没有满秩的假设, 则 $\boldsymbol{x}_{\mathrm{LS}}$ 甚至不是数据的连续函数, $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的微小变化会引起 $\boldsymbol{x}_{\mathrm{LS}}=\boldsymbol{A}^+\boldsymbol{b}$ 的任意大的变化. 能揭示这一点的最容易的办法是考虑广义逆的行为. 如果 $\boldsymbol{A}$ 和 $\delta\boldsymbol{A}$ 属于 $\mathbb{R}^{m\times n}$, 则 Wedin(1973) 和 Stewart(1975) 证明了

$$\|(\boldsymbol{A}+\delta\boldsymbol{A})^+-\boldsymbol{A}^+\|_F\leqslant 2\|\delta\boldsymbol{A}\|_F\max\{\|\boldsymbol{A}^+\|_2^2,\|(\boldsymbol{A}+\delta\boldsymbol{A})^+\|_2^2\}.$$

这个不等式是对定理 2.3.4 的推广, 在定理 2.3.4 中逆矩阵的扰动被限界. 然而, 与非奇异方阵的情形不同的是, 当 $\delta\boldsymbol{A}$ 趋于零时, 上界却不一定趋于零. 如果

$$\boldsymbol{A}=\begin{bmatrix}1&0\\1&0\\0&0\end{bmatrix},\quad \delta\boldsymbol{A}=\begin{bmatrix}0&0\\0&\varepsilon\\0&0\end{bmatrix},$$

则

$$\boldsymbol{A}^+=\begin{bmatrix}1&0&0\\0&0&0\end{bmatrix},\quad (\boldsymbol{A}+\delta\boldsymbol{A})^+=\begin{bmatrix}1&0&0\\1&1/\varepsilon&0\end{bmatrix},$$

且 $\|A^+ - (A+\delta A)^+\|_2 = 1/\varepsilon$. 在有这样的不连续的情形下求 LS 问题的极小数值解是一个很大的挑战.

### 5.5.6 列选主的 QR 分解和基本解

假定 $A \in \mathbb{R}^{m\times n}$ 的秩为 $r$, 列选主的 QR 分解 (算法 5.4.1) 产生分解 $A\Pi = QR$, 其中

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} \begin{matrix} r \\ m-r \end{matrix}.$$
$$\qquad\quad \begin{matrix} r & \quad n-r \end{matrix}$$

有了这个约化, LS 问题就马上解决了. 事实上, 对于任意 $x \in \mathbb{R}^n$, 有

$$\begin{aligned} \|Ax - b\|_2^2 &= \|(Q^{\mathrm{T}}A\Pi)(\Pi^{\mathrm{T}}x) - (Q^{\mathrm{T}}b)\|_2^2 \\ &= \|R_{11}y - (c - R_{12}z)\|_2^2 + \|d\|_2^2 \end{aligned}$$

其中

$$\Pi^{\mathrm{T}}x = \begin{bmatrix} y \\ z \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}, \quad Q^{\mathrm{T}}b = \begin{bmatrix} c \\ d \end{bmatrix} \begin{matrix} r \\ m-r \end{matrix}.$$

于是, 如 $x$ 是一个 LS 问题的极小解, 则必定有

$$x = \Pi \begin{bmatrix} R_{11}^{-1}(c - R_{12}z) \\ z \end{bmatrix}.$$

如果把表达式中的 $z$ 置零, 便可得到**基本解**:

$$x_B = \Pi \begin{bmatrix} R_{11}^{-1}c \\ 0 \end{bmatrix}.$$

注意 $x_B$ 至多有 $r$ 个非零分量, 故 $Ax_B$ 只涉及 $A$ 的列向量的一个子集.

除非子矩阵 $R_{12}$ 是零, 否则这个基本解一般不是极小 2 范数解, 这是因为

$$\|x_{\mathrm{LS}}\| = \min_{z\in\mathbb{R}^{n-r}} \left\| x_B - \Pi \begin{bmatrix} R_{11}^{-1}R_{12} \\ -I_{n-r} \end{bmatrix} z \right\|_2. \tag{5.5.4}$$

事实上, $\|x_{\mathrm{LS}}\|_2$ 的这个特征可用来证明:

$$1 \leqslant \frac{\|x_B\|_2}{\|x_{\mathrm{LS}}\|_2} \leqslant \sqrt{1 + \|R_{11}^{-1}R_{12}\|_2^2}. \tag{5.5.5}$$

详情请参阅 Golub and Pereyra(1976).

### 5.5.7 用 $A\Pi = QR$ 确定数值秩

若用算法 5.4.1 来计算 $x_B$, 那么在确定 $\mathrm{rank}(A)$ 时要非常谨慎. 为了了解这个困难, 假设

$$fl(H_k \cdots H_1 A \Pi_1 \cdots \Pi_k) = \hat{R}^{(k)} = \begin{bmatrix} \hat{R}_{11}^{(k)} & \hat{R}_{12}^{(k)} \\ 0 & \hat{R}_{22}^{(k)} \end{bmatrix} \begin{matrix} k \\ m-k \end{matrix}$$
$$\qquad\qquad\qquad\qquad \begin{matrix} k & \quad n-k \end{matrix}$$

是用浮点运算执行算法的第 $k$ 步之后的矩阵. 假定 $\mathrm{rank}(\boldsymbol{A})=k$. 因为有舍入误差, $\hat{\boldsymbol{R}}_{22}^{(k)}$ 不会精确为零. 但是, 如果 $\hat{\boldsymbol{R}}_{22}^{(k)}$ 的范数适当小, 那么就有理由停止约化, 并宣称 $\boldsymbol{A}$ 的秩为 $k$. 一个有代表性的终止准则是

$$\|\hat{\boldsymbol{R}}_{22}^{(k)}\|_2 \leqslant \varepsilon_1\|\boldsymbol{A}\|_2, \tag{5.5.6}$$

其中 $\varepsilon_1$ 是与机器有关的一个小参数. 根据 Householder 矩阵计算的舍入性质 (见 5.1.12 节), 我们知道 $\hat{\boldsymbol{R}}^{(k)}$ 是 $\boldsymbol{A}+\boldsymbol{E}_k$ 的精确 QR 分解之 $\boldsymbol{R}$, 其中

$$\|\boldsymbol{E}_k\|_2 \leqslant \varepsilon_2\|\boldsymbol{A}\|_2, \quad \varepsilon_2=\boldsymbol{O}(\mathbf{u}).$$

由定理 2.5.2 我们有

$$\sigma_{k+1}(\boldsymbol{A}+\boldsymbol{E}_k)=\sigma_{k+1}(\hat{\boldsymbol{R}}^{(k)}) \leqslant \|\hat{\boldsymbol{R}}_{22}^{(k)}\|_2.$$

又因 $\sigma_{k+1}(\boldsymbol{A}) \leqslant \sigma_{k+1}(\boldsymbol{A}+\boldsymbol{E}_k)+\|\boldsymbol{E}_k\|_2$, 因此

$$\sigma_{k+1}(\boldsymbol{A}) \leqslant (\varepsilon_1+\varepsilon_2)\|\boldsymbol{A}\|_2.$$

换句话说, $\boldsymbol{A}$ 中 $\boldsymbol{O}(\varepsilon_1+\varepsilon_2)$ 的相对扰动能产生一个秩为 $k$ 的矩阵. 以此为终止准则, 可得出结论: 若对某个 $k<n, \hat{\boldsymbol{R}}_{22}^{(k)}$ 适当小时, 列选主的 QR 约化就"发现"了秩亏损.

不幸的是, 情况并非总如此, 一个矩阵可以是几乎秩亏损的, 但没有一个 $\hat{\boldsymbol{R}}_{22}^{(k)}$ 特别小. 于是, 列选主的 QR 法单独作为判断几乎秩亏损的方法不是绝对可靠. 然而, 如果对 $\boldsymbol{R}$ 应用一个好的条件估计数, 在实际中接近秩亏损的情形就不可能不被注意到.

**例 5.5.1**　设矩阵 $\boldsymbol{T}_n(c)$ 如下

$$\boldsymbol{T}_n(c)=\mathrm{diag}(1,s,\cdots,s^{n-1})\begin{bmatrix} 1 & -c & -c & \cdots & -c \\ 0 & 1 & -c & \cdots & -c \\ & \ddots & & \vdots & \vdots \\ \vdots & & & 1 & -c \\ 0 & & \cdots & 0 & 1 \end{bmatrix}$$

其中 $c^2+s^2=1, c,s>0$(参见 Lawson and Hanson(1974, 31 页)). 这些矩阵不会被算法 5.4.1 改变, 因此对 $k=1:n-1$ 有 $\|\boldsymbol{R}_{22}^{(k)}\|_2 \geqslant s^{n-1}$. 这个不等式表明 (举例), 对于矩阵 $T_{100}(0.2)$ 来说, 由于 $s^{99}\approx 0.13$, 它没有特别小的主子矩阵. 然而可以证明 $\sigma_n=O(10^{-8})$.

### 5.5.8　数值秩与 SVD

现在我们集中注意力于有舍入情况时用 SVD 方法处理秩亏损的能力. 回想到如果 $\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$ 是 $\boldsymbol{A}$ 的 SVD, 则

$$\boldsymbol{x}_{\mathrm{LS}}=\sum_{i=1}^{r}\frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\boldsymbol{v}_i \tag{5.5.7}$$

其中 $r=\mathrm{rank}(\boldsymbol{A})$. 用 $\hat{\boldsymbol{U}},\hat{\boldsymbol{V}}$ 和 $\hat{\boldsymbol{\Sigma}}=\mathrm{diag}(\hat{\sigma}_i)$ 表示 $\boldsymbol{U},\boldsymbol{V}$ 和 $\boldsymbol{\Sigma}=\mathrm{diag}(\sigma_i)$ 的计算值. 假定两个奇异值序列都是从大到小排列的. 合理地利用 SVD 算法可推出

$$\hat{\boldsymbol{U}} = \boldsymbol{W} + \Delta \boldsymbol{U}, \quad \boldsymbol{W}^{\mathrm{T}}\boldsymbol{W} = \boldsymbol{I}_m, \quad \|\Delta \boldsymbol{U}\|_2 \leqslant \varepsilon, \tag{5.5.8}$$

$$\hat{\boldsymbol{V}} = \boldsymbol{Z} + \Delta \boldsymbol{V}, \quad \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z} = \boldsymbol{I}_n, \quad \|\Delta \boldsymbol{V}\|_2 \leqslant \varepsilon, \tag{5.5.9}$$

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{W}^{\mathrm{T}}(\boldsymbol{A} + \Delta \boldsymbol{A})\boldsymbol{Z}, \quad \|\Delta \boldsymbol{V}\|_2 \leqslant \varepsilon\|\boldsymbol{A}\|_2, \tag{5.5.10}$$

其中 $\varepsilon$ 是 $\mathbf{u}$ 的一个小倍数, $\mathbf{u}$ 是机器精度. 简言之, SVD 算法将算出“附近的”矩阵 $\boldsymbol{A} + \Delta \boldsymbol{A}$ 的奇异值.

注意 $\hat{\boldsymbol{U}}$ 和 $\hat{\boldsymbol{V}}$ 不一定逼近精确的 $\boldsymbol{U}$ 和 $\boldsymbol{V}$. 但是, 我们可以证明 $\hat{\sigma}_k$ 逼近 $\sigma_k$. 用 (5.5.10) 和定理 2.5.2, 我们有

$$\begin{aligned}\sigma_k &= \min_{\mathrm{rank}(\boldsymbol{B})=k-1} \|\boldsymbol{A} - \boldsymbol{B}\|_2 \\ &= \min_{\mathrm{rank}(\boldsymbol{B})=k-1} \|(\hat{\boldsymbol{\Sigma}} - \boldsymbol{B}) - \boldsymbol{W}^{\mathrm{T}}(\Delta \boldsymbol{A})\boldsymbol{Z}\|_2.\end{aligned}$$

因为 $\|\boldsymbol{W}^{\mathrm{T}}(\Delta \boldsymbol{A})\boldsymbol{Z}\|_2 \leqslant \varepsilon\|\boldsymbol{A}\|_2 = \varepsilon\sigma_1$ 且

$$\min_{\mathrm{rank}(\boldsymbol{B})=k-1} \|\hat{\boldsymbol{\Sigma}}_k - \boldsymbol{B}\|_2 = \hat{\sigma}_k,$$

由此可得, 对 $k = 1:n$ 有 $|\sigma_k - \hat{\sigma}_k| \leqslant \varepsilon\sigma_1$. 于是如果 $\boldsymbol{A}$ 的秩为 $r$, 那么计算得的奇异值中的 $n-r$ 个值会很小. 当计算了 $\boldsymbol{A}$ 的 SVD 时, $\boldsymbol{A}$ 的几乎秩亏损性必定会暴露出来.

**例 5.5.2** 对于例 5.5.1 中的矩阵 $\boldsymbol{T}_{100(0.2)}, \sigma_n \approx 0.367 \times 10^{-8}$.

由计算得的奇异值来估计 $r = \mathrm{rank}(\boldsymbol{A})$ 的一种方法是取参数 $\delta > 0$, 当 $\hat{\sigma}_i$ 满足

$$\hat{\sigma}_1 \geqslant \cdots \geqslant \hat{\sigma}_r \geqslant \delta \geqslant \hat{\sigma}_{r+1} \geqslant \cdots \geqslant \sigma_n$$

时, 我们约定 $\boldsymbol{A}$ 的“数值秩”为 $\hat{r}$. 参数 $\delta$ 应当与机器精度相协调, 例如取 $\delta = \mathbf{u}\|\boldsymbol{A}\|_\infty$. 可是, 如果数据普遍的相对误差比 $\mathbf{u}$ 大, 那么 $\delta$ 也应相应地大一些. 例如, 若 $\boldsymbol{A}$ 的元素只有两位有效数字, 则可取 $\delta = 10^{-2}\|\boldsymbol{A}\|_\infty$.

如果将 $\hat{r}$ 做为数值秩, 我们可将

$$\boldsymbol{x}_{\hat{r}} = \sum_{i=1}^{\hat{r}} \frac{\hat{\boldsymbol{u}}_i^{\mathrm{T}}\boldsymbol{b}}{\hat{\sigma}_i}\hat{\boldsymbol{v}}_i$$

视作 $\boldsymbol{x}_{\mathrm{LS}}$ 的近似解. 由于 $\|\boldsymbol{x}_{\hat{r}}\|_2 \approx 1/\sigma_{\hat{r}} \leqslant 1/\delta$, 也可以按需要选取 $\delta$, 以便在范数适当小的意义下产生近似的 LS 解. 在 12.1 节, 我们将讨论解决此问题的更精致的方法.

如果 $\hat{\sigma}_{\hat{r}} \gg \delta$, 那么我们有理由满足于 $x_{\hat{r}}$, 因为此时可以毫无疑义地视 $\boldsymbol{A}$ 为秩 $\hat{r}$ 矩阵 (对于模 $\delta$).

反过来, $\{\hat{\sigma}_1, \cdots, \hat{\sigma}_n\}$ 不能明显地划分为小的和大的奇异值子集, 这样用该方法确定 $\hat{r}$ 有一些任意性. 这将导致估计秩的更复杂的方法产生, 下面我们针对最小二乘来讨论它们.

例如, 设 $r = n$ 且先假设 (5.5.10) 中的 $\Delta \boldsymbol{A} = 0$. 于是 $\sigma_i = \hat{\sigma}_i (i = 1:n)$. 用 $u_i, w_i, v_i$ 和 $z_i$ 分别表示矩阵 $\hat{\boldsymbol{U}}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{V}}$ 和 $\boldsymbol{Z}$ 的第 $i$ 列. 从 $x_{\mathrm{LS}}$ 减去 $x_{\hat{r}}$ 并取范数得到

$$\|\boldsymbol{x}_{\hat{r}}-\boldsymbol{x}_{\mathrm{LS}}\|_2 \leqslant \sum_{i=1}^{\hat{r}} \frac{\|(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{b})\boldsymbol{z}_i-(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b})\boldsymbol{v}_i\|_2}{\sigma_i}+\sqrt{\sum_{i=\hat{r}+1}^{n}\left(\frac{\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\right)^2}.$$

由 (5.5.8) 和 (5.5.9) 容易证实

$$\|(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{b})\boldsymbol{z}_i-(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b})\boldsymbol{v}_i\|_2 \leqslant 2(1+\varepsilon)\varepsilon\|\boldsymbol{b}\|_2, \tag{5.5.11}$$

因此

$$\|\boldsymbol{x}_{\hat{r}}-\boldsymbol{x}_{\mathrm{LS}}\|_2 \leqslant \frac{\hat{r}}{\sigma_{\hat{r}}}2(1+\varepsilon)\varepsilon\|\boldsymbol{b}\|_2+\sqrt{\sum_{i=\hat{r}+1}^{n}\left(\frac{\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\right)^2}.$$

参数 $\hat{r}$ 可取为使上界最小的整数. 注意上界中的第一项随着 $\hat{r}$ 增大而增大, 而第二项随着 $\hat{r}$ 的增大却是减小的.

当极小化余量比解的精度更重要时, 可以基于我们所推测 $\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_{\hat{r}}\|_2$ 与真的极小值的靠近程度来确定 $\hat{r}$. 和上述分析平行地, 可以证明:

$$\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_{\hat{r}}\|_2-\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_{\mathrm{LS}}\|_2 \leqslant (n-\hat{r})\|\boldsymbol{b}\|_2+\varepsilon\|\boldsymbol{b}\|_2\left(\hat{r}+\frac{\hat{\sigma}_1}{\hat{\sigma}_{\hat{r}}}(1+\varepsilon)\right).$$

同样, 还可选取 $\hat{r}$ 使上界极小. 实际细节请参阅 Varah(1973), 也可见 LAPACK 手册.

### 5.5.9 一些比较

正如我们所指出的, 当用 SVD 解 LS 问题时, 只有 $\boldsymbol{\Sigma}$ 和 $\boldsymbol{V}$ 必须计算. 表 5.5.1 将此方法的效率与其他算法做了比较.

**表 5.5.1**

| LS 算法 | flop 数 |
|---|---|
| 法方程 | $mn^2+n^3/3$ |
| Householder 正交化 | $2mn^2-2n^3/3$ |
| 修正 Gram-Schmidt | $2mn^2$ |
| Givens 正交化 | $3mn^2-n^3$ |
| Householder 双对角化 | $4mn^2-4n^3/2$ |
| R 双对角化 | $2mn^2+2n^3$ |
| Golub-Reinsch SVD | $4mn^2+8n^3$ |
| R-SVD | $2mn^2+11n^3$ |

## 习 题

**5.5.1** 证明如果

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{T} & \boldsymbol{S}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{matrix}r\\ m-r\end{matrix}$$
$$\qquad\quad r \quad n-r$$

有 $r=\mathrm{rank}(\boldsymbol{A})$ 且 $\boldsymbol{T}$ 是非奇异的, 则

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{T}^{-1} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{matrix}r\\ n-r\end{matrix}$$
$$\qquad\quad r \quad m-r$$

满足 $AXA = A$ 和 $(AX)^{\mathrm{T}} = (AX)$. 在这种情况下, 称 $X$ 是 $A$ 的 (1, 3) 广义逆. 证明对于一般的 $A$ 有 $x_B = Xb$, 其中 $X$ 是 $A$ 的 (1,3) 广义逆.

**5.5.2** 定义 $B(\lambda) \in \mathbb{R}^{n\times m}$ 为 $B(\lambda) = (A^{\mathrm{T}}A + \lambda I)^{-1}A^{\mathrm{T}}$, 其中 $\lambda > 0$, 证明

$$\|B(\lambda) - A^{+}\|_2 = \frac{\lambda}{\sigma_r(A)[\sigma_r(A)^2 + \lambda]}, \quad r = \operatorname{rank}(A),$$

从而当 $\lambda \to 0$ 时有 $B(\lambda) \to A^{+}$.

**5.5.3** 考虑秩亏损的 LS 问题

$$\min_{\substack{y\in\mathbb{R}^r \\ z\in\mathbb{R}^{n-r}}} \left\| \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} - \begin{bmatrix} c \\ d \end{bmatrix} \right\|_2,$$

其中 $R \in \mathbb{R}^{r\times r}, S \in \mathbb{R}^{r\times(n-r)}, y \in \mathbb{R}^r, z \in \mathbb{R}^{n-r}$. 假定 $R$ 是上三角形矩阵且非奇异. 说明如何通过不选主元的 QR 分解求得此问题的极小范数解, 并求适当的 $y$ 和 $z$.

**5.5.4** 证明如果有 $A_k \to A$ 和 $A_k^{+} \to A^{+}$ 成立, 则存在一个整数 $k_0$ 满足对所有 $k \geqslant k_0, \operatorname{rank}(A_k)$ 是常数.

**5.5.5** 证明如果 $A \in \mathbb{R}^{m\times n}$ 的秩为 $n$, 则只要 $\|E\|_2\|A^{+}\|_2 < 1$ 就知 $A + E$ 的秩也为 $n$.

## 本节注释与参考文献

伪逆方面的文献浩如烟海, 以下专著列出了 1775 篇文献便是很好的证明:

M. Z. Nashed(1976). *Generalized Inverses and Applications*, Academic Press, New York.

伪逆的微分在下述文章中进一步讨论:

C. L. Lawson and R. J. Hanson(1969). "Extensions and Applications of the Householder Algorithm for Solving Linear Least Squares Problems," *Math. Comp.* 23, 787–812.

G. H. Golub and V. Pereyra(1973). "The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate," *SIAM J. Num. Anal.* 10, 413–432.

LS 问题的扰动理论的综述见于 Lawson and Hanson(1974). Stewart and Sun(1991), Björck (1996) 和

P. A. Wedin(1973). "Perturbation Theory for Pseudo-Inverses," *BIT* 13, 217–232.

G. W. Stewart (1977). "On the Perturbation of Pseudo-Inverses, Projections, and Linear Least Squares," *SIAM Review* 19, 634–662.

即使在满秩情形时, 列选主也能产生更加精确的结果, 下面文章的误差分析给出了原因:

L. S. Jennings and M. R. Osborne(1974). "A. Direct Error Analysis for Least Squares," *Numer. Math.* 22, 322–332.

下述文章对秩亏损情形做了讨论:

J. M. Varah(1973). "On the Numerical Solution of Ill-Conditioned Linear Systems with Applications to Ill-Posed Problems," *SIAM J. Num. Anal.* 10, 257–267.

G. W. Stewart(1984). "Pank Degeneracy," *SIAM J. Sci. and Stat. Comp.* 5, 403–413.

P. C. Hansen(1987). "The Truncated SVD as a Method for Regularization," *BIT* 27, 534–553.

G. W. Stewart(1987). "Collinearity and Least Squares Regression," *Statistical Science* 2, 68–100.

在 12.1 节和 12.2 节我们就这个主题还要做更多讨论.

## 5.6 加权和迭代改进

在第 3 章中关于方线性方程组引进了加权和迭代改进的概念. 现在, 将把这些概念推广到最小二乘问题.

### 5.6.1 列加权

假定 $\boldsymbol{G} \in \mathbb{R}^{n\times n}$ 是非奇异的. LS 问题

$$\min \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2, \quad \boldsymbol{A} \in \mathbb{R}^{m\times n},\ \boldsymbol{b} \in \mathbb{R}^m \tag{5.6.1}$$

的解可以通过先求

$$\min \|(\boldsymbol{A}\boldsymbol{G})\boldsymbol{y} - \boldsymbol{b}\|_2 \tag{5.6.2}$$

的极小 2 范数 $\boldsymbol{y}_{\text{LS}}$, 然后令 $\boldsymbol{x}_G = \boldsymbol{G}\boldsymbol{y}_{\text{LS}}$ 而得到. 如果 $\text{rank}(\boldsymbol{A}) = n$, 那么 $\boldsymbol{x}_G = \boldsymbol{x}_{\text{LS}}$. 否则, $x_G$ 是 (5.6.1) 的极小 $\boldsymbol{G}$ 范数解, 其中 $\boldsymbol{G}$ 范数定义为 $\|\boldsymbol{z}\|_G = \|\boldsymbol{G}^{-1}\boldsymbol{z}\|_2$.

$\boldsymbol{G}$ 的选取是重要的. 有时它的选择可基于对 $\boldsymbol{A}$ 的不确定因素的先验估计. 另一些时候, 可取

$$\boldsymbol{G} = \boldsymbol{G}_0 \equiv \text{diag}(1/\|A(:,1)\|_2, \cdots, 1/\|A(:,n)\|_2)$$

来使 $\boldsymbol{A}$ 的列规范化. Van der Sluis(1969) 证明了对此选取, $\kappa_2(\boldsymbol{A}\boldsymbol{G})$ 近似极小. 由于 $y_{\text{LS}}$ 的计算精度依赖于 $\kappa_2(\boldsymbol{A}\boldsymbol{G})$, 取 $\boldsymbol{G} = \boldsymbol{G}_0$ 是有道理的.

我们发现列加权会改变奇异值. 因而当确定数值秩的方法用于 $\boldsymbol{A}$ 和 $\boldsymbol{A}\boldsymbol{G}$ 时可能不会给出同样的估计. 见 Stewart(1984 b).

### 5.6.2 行加权

设 $\boldsymbol{D} = \text{diag}(d_1, \cdots, d_m)$ 是非奇异的, 考虑*加权最小二乘问题*:

$$\min \|\boldsymbol{D}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\|_2, \quad \boldsymbol{A} \in \mathbb{R}^{m\times n}, \quad \boldsymbol{b} \in \mathbb{R}^m. \tag{5.6.3}$$

假定 $\text{rank}(A) = n$ 且 $\boldsymbol{x}_D$ 是 (5.6.3) 的解. 可证 (5.6.1) 的解 $\boldsymbol{x}_{\text{LS}}$ 满足

$$\boldsymbol{x}_D - \boldsymbol{x}_{\text{LS}} = (\boldsymbol{A}^{\text{T}}\boldsymbol{D}^2\boldsymbol{A})^{-1}\boldsymbol{A}^{\text{T}}(\boldsymbol{D}^2 - \boldsymbol{I})(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{\text{LS}}) \tag{5.6.4}$$

这表明 LS 问题的行加权会改变解 (一个重要例外是 $\boldsymbol{b} \in \text{ran}(\boldsymbol{A})$, 此时 $\boldsymbol{x}_D = \boldsymbol{x}_{\text{LS}}$).

确定 $\boldsymbol{D}$ 的一种方法是令 $d_k$ 是 $b_k$ 的不确定性的某种量度, 例如, $b_k$ 的标准偏差的倒数. 这样, 当 $d_k$ 较大时, $r_k = \boldsymbol{e}_k^{\text{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_D)$ 倾向于变小. $d_k$ 对 $r_k$ 的精确影响可阐明如下. 定义

$$\boldsymbol{D}(\delta) = \text{diag}(d_1, \cdots, d_{k-1}, d_k\sqrt{1+\delta}, d_{k+1}, \cdots, d_m),$$

其中 $\delta > -1$. 如果 $x(\delta)$ 极小化 $\|\boldsymbol{D}(\delta)(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\|_2$ 且 $r_k(\delta)$ 是 $\boldsymbol{b} - \boldsymbol{A}x(\delta)$ 的第 $k$ 个分量, 则可证:

$$r_k(\delta) = \frac{r_k}{1 + \delta d_k^2 e_k^{\text{T}}\boldsymbol{A}(\boldsymbol{A}^{\text{T}}\boldsymbol{D}^2\boldsymbol{A})^{-1}\boldsymbol{A}^{\text{T}}\boldsymbol{e}_k}. \tag{5.6.5}$$

这个显式表达式说明 $r_k(\delta)$ 是 $\delta$ 的单调递减函数. 当然, 当所有的权都变化时 $r_k$ 如何变化要比这复杂得多.

**例 5.6.1** 假定

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

如果 $D = I_4$, 则 $x_D = [-1, 0.85]^T$ 和 $r = b - Ax_D = [0.3, -0.4, -0.1, 0.2]^T$. 另一方面, 如果 $D = \text{diag}(1000, 1, 1, 1)$, 则我们有 $x_D \approx [-1.43, 1.21]^T$ 和 $r = b - Ax_D = [0.000\,428, -0.571\,428, -0.142\,853, 0.285\,714]^T$.

### 5.6.3 广义最小二乘

在许多估计问题中, 观测向量 $b$ 与 $x$ 之间的关系为

$$b = Ax + w, \tag{5.6.6}$$

其中*噪声向量* $w$ 的均值为零, 而且其*方差–协方差矩阵* $\sigma^2 W$ 对称正定. 假定已知 $W$ 且对某 $B \in \mathbb{R}^{m\times m}$ 有 $W = BB^T$. $B$ 可能是给定的也可能是 $W$ 的 Cholesky 三角形矩阵. 为了使 (5.6.6) 中的每一个方程对于确定的 $x$ 起同等作用, 统计学家们常常要解 LS 问题

$$\min \|B^{-1}(Ax - b)\|_2. \tag{5.6.7}$$

对此问题的一个显然的计算方法是先形成 $\tilde{A} = B^{-1}A$ 和 $\tilde{b} = B^{-1}b$, 然后用我们已介绍过的任何方法极小化 $\|\tilde{A}\,x - \tilde{b}\|_2$. 不幸的是, 当 $B$ 是病态的时, 这种方式得到的 $x$ 稳定性非常差.

Paige(1979a, 1979b) 提出一个稳定得多的求解 (5.6.7) 的方法. 它利用正交变换, 基于 (5.6.7) 与以下*广义最小二乘*问题

$$\min_{b = Ax + Bv} v^T v \tag{5.6.8}$$

的等价性. 注意到即使 $A$ 和 $B$ 都是秩亏损的, (5.6.8) 也是有定义的. 虽然 Paige 的方法对此情况可以应用, 但我们描述它时仍假定这两个矩阵都是满秩的.

第一步是计算 $A$ 的 QR 分解

$$Q^T A = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \qquad Q = [\underset{n}{Q_1} \quad \underset{m-n}{Q_2}].$$

然后, 确定正交矩阵 $Z \in \mathbb{R}^{m\times m}$ 使得

$$Q_2^T B Z = [\underset{n}{0} \quad \underset{m-n}{S}], \qquad Z = [\underset{n}{Z_1} \quad \underset{m-n}{Z_2}],$$

其中 $S$ 是上三角形矩阵. 利用这些正交矩阵可把 (5.6.8) 中的约束条件化成

$$\begin{bmatrix} Q_1^T b \\ Q_2^T b \end{bmatrix} = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} x + \begin{bmatrix} Q_1^T B Z_1 & Q_1^T B Z_2 \\ 0 & S \end{bmatrix} \begin{bmatrix} Z_1^T v \\ Z_2^T v \end{bmatrix}.$$

此方程的“下半部”可以解出 $\boldsymbol{v}$,

$$\boldsymbol{S}\boldsymbol{u} = \boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{b}, \quad \boldsymbol{v} = \boldsymbol{Z}_2\boldsymbol{u}, \tag{5.6.9}$$

而它的“上半部”确定了 $\boldsymbol{x}$:

$$\begin{aligned}\boldsymbol{R}_1\boldsymbol{x} &= \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{b} - (\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}_1\boldsymbol{Z}_1^{\mathrm{T}} + \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}_2\boldsymbol{Z}_2^{\mathrm{T}})\boldsymbol{v} \\ &= \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{b} - \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}_2\boldsymbol{u}.\end{aligned} \tag{5.6.10}$$

这个方法吸引人之处是所有潜在的病态都集中在三角形方程组 (5.6.9) 和 (5.6.10). 此外, Paige(1979b) 证明了上述的方法是数值稳定的, 这是任何显式地形成 $\boldsymbol{B}^{-1}\boldsymbol{A}$ 的方法所不具有的.

### 5.6.4 迭代改进

Björck(1967, 1968) 分析了改善 LS 近似解的技术. 它基于如下思想：如果

$$\begin{bmatrix} \boldsymbol{I}_m & \boldsymbol{A} \\ \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{x} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{A} \in \mathbb{R}^{m\times n}, \quad \boldsymbol{b} \in \mathbb{R}^m, \tag{5.6.11}$$

则 $\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2 = \min$. 这是因为从 $\boldsymbol{r} + \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{r} = \boldsymbol{0}$ 可推出 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$. 如果 $\mathrm{rank}(\boldsymbol{A}) = n$( 我们下面就假设如此), 则上面的增广方程组是非奇异的.

把 LS 问题化为方线性方程组后, 可应用迭代改进格式 (3.5.5).

$r^{(0)} = 0; x^{(0)} = 0$

**for** $k = 0, 1,$

$$\begin{bmatrix} f^{(k)} \\ g^k \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} - \begin{bmatrix} I & A \\ A^{\mathrm{T}} & 0 \end{bmatrix} \begin{bmatrix} r^{(k)} \\ x^{(k)} \end{bmatrix}$$

$$\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} p^{(k)} \\ z^{(k)} \end{bmatrix} = \begin{bmatrix} f^{(k)} \\ g^{(k)} \end{bmatrix}$$

$$\begin{bmatrix} r^{(k+1)} \\ x^{(k+1)} \end{bmatrix} = \begin{bmatrix} r^{(k)} \\ x^{(k)} \end{bmatrix} + \begin{bmatrix} p^{(k)} \\ z^{(k)} \end{bmatrix}$$

**end**

必须用较高的精度来计算余量 $\boldsymbol{f}^{(k)}$ 和 $\boldsymbol{g}^{(k)}$, 为此要保留原始的 $\boldsymbol{A}$.

如果有了 $\boldsymbol{A}$ 的 QR 分解, 则很容易获得增广方程组的解. 具体地说, 如果 $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$ 和 $\boldsymbol{R}_1 = R(1:n, 1:n)$, 则形如

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{A} \\ \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix}$$

的方程组变换为

$$\begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{0} & \boldsymbol{R}_1 \\ \boldsymbol{0} & \boldsymbol{I}_{m-n} & \boldsymbol{0} \\ \boldsymbol{R}_1^{\mathrm{T}} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{h} \\ \boldsymbol{f}_2 \\ \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \boldsymbol{g} \end{bmatrix},$$

其中

$$Q^{\mathrm{T}}f=\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}\begin{matrix} n \\ m-n \end{matrix}, \quad Q^{\mathrm{T}}p=\begin{bmatrix} h \\ f_2 \end{bmatrix}\begin{matrix} n \\ m-n \end{matrix}.$$

因此, 通过解三角形方程组 $R_1^{\mathrm{T}}h=g$ 和 $R_1z=f_1-h$ 并令 $p=Q\begin{bmatrix} h \\ f_2 \end{bmatrix}$ 就可得到 $p$ 和 $z$. 假定 $Q$ 是以因子形式存储的, 那么每一次迭代需要 $8mn-2n^2$ 个 flop.

使迭代成功的关键在于同时修正 LS 余量和解, 而不仅仅是解本身. Björck (1968) 证明, 如果 $\kappa(A)\approx\beta^q$ 且用 $\beta$ 进位 $t$ 位有效数字运算, 那么只要用双精度计算余量, 则 $x^{(k)}$ 约有 $k(t-q)$ 位准确数字. 注意这个直观分析中出现的是 $\kappa_2(A)$ 而不是 $\kappa_2(A)^2$.

## 习 题

**5.6.1** 证明公式 (5.6.4).

**5.6.2** 设 $A\in\mathbb{R}^{m\times n}$ 是满秩的, 对 $\delta>-1$ 定义对角矩阵

$$\Delta=\mathrm{diag}(\underbrace{1,\cdots,1}_{k-1},(1+\delta),\underbrace{1,\cdots,1}_{m-k}),$$

用 $x(\delta)$ 来表示 $\min\|\Delta(Ax-b)\|_2$ 的 LS 解, 用 $r(\delta)=b-Ax(\delta)$ 来表示其余量.

(a) 证明 $r(\delta)=\left(I-\delta\dfrac{A(A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}e_ke_k^{\mathrm{T}}}{1+\delta e_k^{\mathrm{T}}A(A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}e_k}\right)r(0)$.

(b) 令 $r_k(\delta)$ 表示 $r(\delta)$ 的第 $k$ 个元素, 证明

$$r_k(\delta)=\frac{r_k(0)}{1+\delta e_k^{\mathrm{T}}A(A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}e_k}.$$

(c) 利用 (b) 去证明公式 (5.6.5).

**5.6.3** 在公式 (5.6.8) 中当 $A$ 和 $B$ 是秩亏损时, 考虑如何用 SVD 法求解广义 LS 问题

**5.6.4** 设 $A\in\mathbb{R}^{m\times n}$ 的秩为 $n$, 对于 $\alpha\geqslant 0$, 定义

$$M(\alpha)=\begin{bmatrix} \alpha I_m & A \\ A^{\mathrm{T}} & 0 \end{bmatrix}.$$

证明

$$\sigma_{m+n}(M(\alpha))=\min\left\{\alpha,-\frac{\alpha}{2}+\sqrt{\sigma_n(A)^2+\left(\frac{\alpha}{2}\right)^2}\right\},$$

并确定使 $K_2(M(\alpha))$ 取极小值的 $\alpha$.

**5.6.5** 另一种 LS 问题的迭代改进方法如下:

$x^{(0)}=0$

**for** $k=0,1,\cdots$

  $r^{(k)}=b-Ax^{(k)}$ (双精度)

  $\|Az^{(k)}-r^{(k)}\|_2=\min$

  $x^{k+1}=x^{(k)}+z^{(k)}$

**end**

(a) 假设已有 $\boldsymbol{A}$ 的 QR 分解, 每次迭代需多少 flop?

(b) 证明在 5.6.4 节中给出的迭代改进方案中, 令 $g^{(k)}=0$, 则得到上面的算法.

### 本节注释与参考文献

LS 问题的行加权和列加权在 Lawson and Hanson(SLS, 第 180~188 页) 中有讨论. 对加权的各种效果之讨论见:

A. van der Sluis(1969). "Condition Numbers and Equilibration of Matrices," *Numer. Math.* 14, 14–23.

G. W. Stewart(1948b). "On the Asymptotic Behavior of Scaled Singular Value and QR Decompositions," *Math. Comp.* 43, 483–490.

广义 LS 问题的理论和计算的性质出自于:

S. Kourouklis and C. C. Paige(1981). "A Constrained Least Squares Approach to the General Gauss-Markov Linear Model," *J. Amer, Stat. Assoc.* 76, 620–625.

C. C. Paige(1979a). "Computer Solution and Perturbation Analysis of Generalized Least Squares Problems," *Math. Comp.* 33, 171–184.

C. C. Paige(1979b). "Fast Numerically Stable Computations for Generalized Linear Least Squares Problems," *SIAM J. Num. Anal.* 16, 165–171.

C. C. Paige(1985). "The General Limit Model and the Generalized Singular Value Decomposition," *Lin. Alg. and Its Applic.* 70, 269–284.

最小二乘的迭代改进见:

G. H. Golub and J. H. Wilkinson(1966). "Note on Iterative Refinement of Least Squares Solutions," *Numer. Math.* 9, 139–148.

Å. Björck and G. H. Golub(1967). "Iterative Refinement of Linear Least Squares Solutions by Householder Transformation," *BIT* 7, 322–337.

Å. Björck(1967). "Iterative Refinement of Linear Least Squares Solutions I," *BIT* 7, 257–278.

Å. Björck(1968). "Iterative Refinement of Linear Least Squares Solutions II," *BIT* 8, 8–30.

Å. Björck(1987). "Stability Analysis of the Method of Seminormal Equations for Linear Least Squares Problems," *Linear Alg. and Its Applic.* 88/89, 31–48.

## 5.7 正方形方程组和欠定方程组

本章的正交化方法可用于正方形方程组, 也可用于方程个数比未知数少的方程组. 在这简短的一节中我们讨论几种可能的情况.

### 5.7.1 用 QR 分解和 SVD 求解正方形方程组

基于 QR 分解和 SVD 的最小二乘法, 可用来求解正方形方程组: 只需令 $m=n$. 可是, 从 flop 数来看, 高斯消去法是解正方形方程组的最经济的方法. 这从表 5.7.1

即可得知, 在此我们假定在分解时右端项是已知的.

尽管如此, 有三个理由说明为什么仍然要考虑正交化方法:

- 仅考虑工作量夸大了高斯消去法的优点. 当考虑内存通信和向量化的开销, QR 分解法的效率毫不逊色.
- 正交化法能保证稳定性, 不必像高斯消去法那样担心"增长因子".
- 对于病态问题, 正交化方法更加可靠. 带条件数估计的 QR 方法是值得依赖的. 当然, 在求几乎奇异方程组的有意义的解方面, SVD 方法是无与伦比的.

我们并不是在表示强烈偏爱正交化方法, 而只是在说它们和高斯消去法各具特色.

**表 5.7.1**

| 方　法 | flop 数 |
|---|---|
| 高斯消去法 | $2n^3/3$ |
| Householder 正交化 | $4n^3/3$ |
| 修正 Gram-Schmidt | $2n^3$ |
| 双对角化 | $8n^3/3$ |
| SVD(奇异值分解) | $12n^3$ |

我们还要指出, 表 5.7.1 中 SVD 对应的值是假定在因子分解时就有 $b$. 否则, 有必要累积 $\boldsymbol{U}$ 矩阵, 工作量则为 $20n^3$ 个 flop.

如果用 QR 分解法解 $\boldsymbol{Ax} = \boldsymbol{b}$, 则通常需要进行向后消去: $\boldsymbol{Rx} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b}$. 然而, 可通过对 $\boldsymbol{b}$ 进行预处理避免此步骤. 假定 $\boldsymbol{H}$ 是一个 Householder 矩阵使得 $\boldsymbol{Hb} = \beta \boldsymbol{e}_n$, 其中 $\boldsymbol{e}_n$ 是 $\boldsymbol{I}_n$ 的最后一列. 如果我们计算 $(\boldsymbol{HA})^{\mathrm{T}}$ 的 QR 分解, 则 $\boldsymbol{A} = \boldsymbol{H}^{\mathrm{T}}\boldsymbol{R}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}}$, 方程组化为

$$\boldsymbol{R}^{\mathrm{T}}y = \beta e_n,$$

其中 $\boldsymbol{y} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x}$. 因为 $\boldsymbol{R}^{\mathrm{T}}$ 是下三角形矩阵, $\boldsymbol{y} = (\beta/r_{nn})\boldsymbol{e}_n$, 故

$$\boldsymbol{x} = \frac{\beta}{r_{nn}}\boldsymbol{Q}(:,n).$$

### 5.7.2 欠定方程组

当 $m < n$ 时, 我们称线性方程组

$$\boldsymbol{Ax} = \boldsymbol{b}, \quad \boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{b} \in \mathbb{R}^{m} \tag{5.7.1}$$

是欠定的. 注意这样的方程组要么无解要么有无穷多个解. 当有无穷多解时, 重要的是区分算法是否找到极小 2 范数解. 我们给出的第一个算法找到的就不一定是极小 2 范数解. 假定 $\boldsymbol{A}$ 是行满秩的, 应用列选主的 QR 方法得到

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A\Pi} = [\boldsymbol{R}_1 \quad \boldsymbol{R}_2]$$

其中 $\boldsymbol{R}_1 = \mathbb{R}^{m\times n}$ 是上三角形矩阵, $\boldsymbol{R}_2 \in \mathbb{R}^{m\times(n-m)}$. 于是 $\boldsymbol{Ax} = \boldsymbol{b}$ 化为

$$(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A\Pi})(\boldsymbol{\Pi}^{\mathrm{T}}\boldsymbol{x}) = [\boldsymbol{R}_1 \quad \boldsymbol{R}_2]\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b},$$

其中

$$\boldsymbol{\Pi}^{\mathrm{T}}\boldsymbol{x} = \begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{bmatrix}$$

且 $\boldsymbol{z}_1 \in \mathbb{R}^m, \boldsymbol{z}_2 \in \mathbb{R}^{(n-m)}$. 由于列选主, 而且我们假定 $\boldsymbol{A}$ 是行满秩的, 所以 $\boldsymbol{R}_1$ 是非奇异的. 于是, 令 $\boldsymbol{z}_1 = \boldsymbol{R}_1^{-1}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b}$ 和 $\boldsymbol{z}_2 = \boldsymbol{0}$ 就得到问题的一个解.

**算法 5.7.1** 给定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \mathrm{rank}(\boldsymbol{A}) = m, \boldsymbol{b} \in \mathbb{R}^m$, 本算法寻找 $\boldsymbol{x} \in \mathbb{R}^n$ 使得 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.

$Q^{\mathrm{T}}A\Pi = R$ (列选主的 QR 分解)

解 $R(1:m, 1:m)z_1 = Q^{\mathrm{T}}b$.

令 $x = \Pi \begin{bmatrix} z_1 \\ 0 \end{bmatrix}$.

此算法需要 $2m^2n - m^3/3$ 个 flop. 不能保证是极小范数解 (一个不同的 $\Pi$ 会得到一个更小的 $\boldsymbol{z}_1$). 但是, 如果我们计算 QR 分解

$$\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{Q}\boldsymbol{R} = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{0} \end{bmatrix},$$

其中 $\boldsymbol{R}_1 \in \mathbb{R}^{m\times m}$, 则 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 化为

$$(\boldsymbol{Q}\boldsymbol{R})^{\mathrm{T}}\boldsymbol{x} = [\boldsymbol{R}_1^{\mathrm{T}} \quad 0]\begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{bmatrix} = \boldsymbol{b},$$

其中

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x} = \begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{bmatrix}, \quad \boldsymbol{z}_1 \in \mathbb{R}^m, \quad \boldsymbol{z}_2 \in \mathbb{R}^{n-m}.$$

现在, 令 $\boldsymbol{z}_2 = \boldsymbol{0}$ 就得到了极小范数解.

**算法 5.7.2** 给定 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \mathrm{rank}(\boldsymbol{A}) = m, \boldsymbol{b} \in \mathbb{R}^m$, 本算法寻找 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的极小 2 范数解.

$\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{Q}\boldsymbol{R}$ (QR 分解)

解 $\boldsymbol{R}(1:m, 1:m)^{\mathrm{T}}z = b$

$x = \boldsymbol{Q}(:, 1:m)z$.

本算法最多需要 $2m^2n - 2m^3/3$ 个 flop.

SVD 也可用来计算欠定 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 问题的极小范数解. 如果

$$\boldsymbol{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathrm{T}}, \quad r = \mathrm{rank}(\boldsymbol{A})$$

是 $\boldsymbol{A}$ 的奇异值展开式, 则

$$\boldsymbol{x} = \sum_{i=1}^{r} \frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\boldsymbol{v}_i.$$

与最小二乘问题一样, 当 $\boldsymbol{A}$ 是几乎秩亏损时, SVD 方法是令人满意的.

### 5.7.3 扰动的欠定方程组

我们以满秩欠定方程组的扰动结果来结束本节.

**定理 5.7.1** 假定 $\text{rank}(\boldsymbol{A}) = m \leqslant n, \boldsymbol{A} \in \mathbb{R}^{m\times n}, \delta\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{0} \neq \boldsymbol{b} \in \mathbb{R}^m$ 满足

$$\varepsilon = \max\{\varepsilon_A, \varepsilon_b\} < \sigma_m(\boldsymbol{A}),$$

其中 $\varepsilon_{\boldsymbol{A}} = \|\delta\boldsymbol{A}\|_2/\|\boldsymbol{A}\|_2, \varepsilon_{\boldsymbol{b}} = \|\delta\boldsymbol{b}\|_2/\|\boldsymbol{b}\|_2$. 如果 $\boldsymbol{x}$ 和 $\hat{\boldsymbol{x}}$ 是满足

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \quad (\boldsymbol{A} + \delta\boldsymbol{A})\hat{\boldsymbol{x}} = \boldsymbol{b} + \delta\boldsymbol{b}$$

的极小范数解, 则

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leqslant \kappa_2(\boldsymbol{A})(\varepsilon_{\boldsymbol{A}} \min\{2, n-m+1\} + \varepsilon_b) + O(\varepsilon^2).$$

**证明** 令 $\boldsymbol{E}$ 和 $\boldsymbol{f}$ 分别为 $\delta\boldsymbol{A}/\varepsilon$ 和 $\delta\boldsymbol{b}/\varepsilon$. 注意, 对所有的 $0 < t < \varepsilon$ 有 $\text{rank}(\boldsymbol{A} + t\boldsymbol{E}) = m$, 而且

$$x(t) = (\boldsymbol{A} + t\boldsymbol{E})^{\mathrm{T}}((\boldsymbol{A} + t\boldsymbol{E})(\boldsymbol{A} + t\boldsymbol{E})^{\mathrm{T}})^{-1}(\boldsymbol{b} + t\boldsymbol{f})$$

满足 $(\boldsymbol{A} + t\boldsymbol{E})x(t) = b + tf$. 将此表达式对 $t$ 求微分, 并在所得结果中令 $t = 0$ 可得

$$\dot{x}(0) = (\boldsymbol{I} - \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}\boldsymbol{A})\boldsymbol{E}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}b + \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}(\boldsymbol{f} - \boldsymbol{E}\boldsymbol{x}).$$

由于

$$\|\boldsymbol{x}\|_2 = \|\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}\boldsymbol{b}\|_2 \geqslant \sigma_m(\boldsymbol{A})\|(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}\boldsymbol{b}\|_2,$$
$$\|\boldsymbol{I} - \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}\boldsymbol{A}\|_2 = \min(1, n-m),$$
$$\frac{\|\boldsymbol{f}\|_2}{\|\boldsymbol{x}\|_2} \leqslant \frac{\|\boldsymbol{f}\|_2\|\boldsymbol{A}\|_2}{\|\boldsymbol{b}\|_2},$$

我们有

$$\frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \frac{x(\varepsilon) - x(0)}{\|x(0)\|_2} = \varepsilon\frac{\|\dot{\boldsymbol{x}}(0)\|_2}{\|\boldsymbol{x}\|_2} + O(\varepsilon^2)$$
$$\leqslant \varepsilon\min(1, n-m)\left\{\frac{\|\boldsymbol{E}\|_2}{\|\boldsymbol{A}\|_2} + \frac{\|\boldsymbol{f}\|_2}{\|\boldsymbol{b}\|_2} + \frac{\|\boldsymbol{E}\|_2}{\|\boldsymbol{A}\|_2}\right\}\kappa_2(\boldsymbol{A}) + O(\varepsilon^2),$$

由此知定理成立. □

注意, 与超定方程组的情形不一样, 这里没有 $\kappa_2(\boldsymbol{A})^2$ 因子.

## 习 题

**5.7.1** 试推出上式中 $\dot{x}(0)$ 的表达式.

**5.7.2** 寻找 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的极小范数解, 其中 $\boldsymbol{A} = [1 \quad 2 \quad 3], \boldsymbol{b} = 1$.

**5.7.3** 说明用 QR 分解解欠定方程组时如何能避免求解三角形方程组.

**5.7.4** 假设 $\boldsymbol{b}, \boldsymbol{x} \in \mathbb{R}^n$ 给定, 考虑下面问题:

(a) 寻找一个非对称 Toeplitz 矩阵 $\boldsymbol{T}$ 使得 $\boldsymbol{T}\boldsymbol{x} = \boldsymbol{b}$;

(b) 寻找一个对称 Toeplitz 矩阵 $\boldsymbol{T}$ 使得 $\boldsymbol{Tx} = \boldsymbol{b}$;

(c) 寻找一个循环矩阵 $\boldsymbol{C}$ 使得 $\boldsymbol{Cx} = \boldsymbol{b}$.

将每个问题写成 $\boldsymbol{Ap} = \boldsymbol{b}$ 的形式, 其中 $\boldsymbol{A}$ 是由 $\boldsymbol{x}$ 的元素组成的矩阵, $\boldsymbol{p}$ 为需要求的元素所组成的向量.

## 本节注释与参考文献

有关奇异方程组的一些有趣的现象见:

T. F. Chan(1984). "Deflated Decomposition Solutions of Nearly Singular Systems," *SIAM J. Num. Anal.* 21, 738–754.

G. H. Golub and C. D. Meyer(1986). "Using the QR Factorization and Group Inversion to Compute, Differentiate, and estimate the Sensitivity of Stationary Probabilities for Markov Chains," *SIAM J. Alg. and Dis. Methods*, 7, 273–281.

关于欠定方程组的文章包括:

R. E. Cline and R. J. Plemmons (1976). "$L_2$-Solutions to Underdetermined Linear Systems," *SIAM Review* 18, 92–106.

M. Arioli and A. Laratta(1985). "Error Analysis of an Algorithm for Solving an Underdetermined System," *Numer. Math.* 46, 255–268.

J. W. Demmel and N. J. Higham(1993). "Improved Error Bounds for Underdetermined System Solvers," *SIAM J. Matrix Anal. Appl.* 14, 1–14.

QR 分解当然可用于求解线性方程组, 见:

N. J. Higham(1991). "Iterative Refinement Enhances the Stability of QR Factorization Methods for Solving Linear Equations," *BIT* 31, 447–468.

# 第6章　并行矩阵计算

并行矩阵计算现已成为研究的热点. 尽管大部分的工作是与机器/系统相关的, 还是涌现了许多基本的技巧. 本章主要介绍这些技巧, 同时描绘在矩阵计算的设计中怎样用 "并行" 去考虑问题.

我们将考虑分布式内存系统和共享内存系统. 在 6.1 节中, 利用矩阵与向量相乘来引出节点程序的概念. 同时讨论了均衡负载, 加速比和同步问题. 6.2 节中矩阵与矩阵相乘显示了矩阵分块运算的效果并给出了二维数据流动的思想. 6.3 节给出了 Cholesky 分解的两个并行实现.

**预备知识**

阅读本章需要掌握第 1 章以及 4.1 节的知识和 4.2 节的知识. 本章各节间的关系如下:

$$6.1\text{节} \to 6.2\text{节} \to 6.3\text{节}$$

补充参考文献包括专著 Schönauer(1987), Hockney and Jesshope(1988), Modi (1988), Ortega (1988), Dongarra, Duff, Sorensen, and van der Vorst (1991), Golub and Ortega (1993) 及综述性文章 Heller (1978), Ortega and Voight (1985), Gallian, Plemmons, and Sameh (1990), Demmel, Heath, and Vander Vorst(1993).

## 6.1　基本概念

在本节, 我们以 gaxpy 运算:

$$z = y + Ax, \quad A \in \mathbb{R}^{n\times n}, \quad x, y, z \in \mathbb{R}^n \tag{6.1.1}$$

为例来介绍分布式内存系统和共享内存系统. 实际上, 这两种类型的并行计算之界限是模糊的, 而且我们的各种讨论之融合常常可适用于任何具体机器.

### 6.1.1　分布式内存系统

在拥有分布式内存的多处理机中, 每个处理器都有*本地内存*并能够执行其*节点程序*. 节点程序可以改变所执行的处理器的本地内存中的值, 还能够以*消息*的形式向网络上其他处理器传送数据. 处理器间的互联定义了*网络拓扑*, 一个简单例子是*环*, 用它做入门介绍是足够的. 见图 6.1.1.

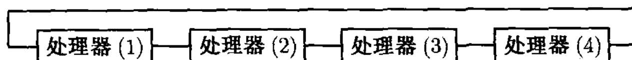



图 6.1.1　四处理器的环

其他一些重要的互联结构包括网结构、环面结构 (与二维数组密切对应)、超立方体 (hypercube) 结构 (普遍性和最优性)、树形结构 (适合于分而治之算法). 关于各种情形的讨论请参阅 Ortega and Voigt(1985). 我们的直接目标是为 (6.1.1) 建立一个基于环的算法. 基于环面的矩阵乘法将在 6.2 节中讨论.

每个处理器都有一个*标识号*, 第 $\mu$ 个处理器记为 $\text{Proc}(\mu)$. 如果 $\text{Proc}(\lambda)$ 和 $\text{Proc}(\mu)$ 之间有一物理线路直接相连, 则称二者*相邻*. 因此, 在有 $p$ 个处理器的环中, $\text{Proc}(p)$ 与 $\text{Proc}(p-1)$ 和 $\text{Proc}(1)$ 相邻.

设计有效的分布式内存算法的重要因素包括: (1) 处理器数和本地内存容量, (2) 处理器间互联方式, (3) 计算速度与处理器间通信速度的比例关系, (4) 节点能否同时计算和通信.

### 6.1.2 通信

用一个简单的表示来描述发收消息:

$$\textbf{send}(\{\text{matrix}\}, \{\text{接收处理器的序号}\})$$

$$\textbf{recv}(\{\text{matrix}\}, \{\text{发送处理器的序号}\})$$

标量和向量都可看作矩阵, 因此都可作为消息. 在此模型中, 如果 $\text{Proc}(\mu)$ 执行指令 **send** $(V_{\text{loc}}, \lambda)$, 则本地矩阵 $V_{\text{loc}}$ 的一个副本被发送到 $\text{Proc}(\lambda)$, $\text{Proc}(\mu)$ 的节点程序被迅速恢复执行. 处理器发送消息给自己也是允许的. 利用下标 loc 来强调一个矩阵被存储在本地内存中.

如果 $\text{Proc}(\mu)$ 执行指令 $\text{recv}(U_{\text{loc}}, \lambda)$, 那么节点程序暂停, 直到收到 $\text{Proc}(\lambda)$ 发来的消息为止. 一旦收到, 消息存于本地矩阵 $U_{\text{loc}}$ 中, $\text{Proc}(\mu)$ 恢复节点程序运行.

尽管这种发送/接收表示法足够我们使用, 但其中还是掩盖了许多重要细节.

- **装配消息开销** *在实际中, 由于一个矩阵的元素有可能在发送者的内存中不是连续存储的, 这会导致多的耗费. 我们忽略其细节.*
- **标记消息开销** *消息不一定按发送的顺序到达. 为使接收者不产生混乱, 就需对每个消息加一个标记. 我们假定消息总是按发送的顺序到达而忽略这方面的细节.*
- **消息解释开销** *在实际中, 一个消息是一组位流, 必须提供一个信头来告诉接收者矩阵的维数和表示矩阵元素的浮点字的格式. 将消息转变为矩阵需要时间, 但我们没有去量化此开销.*

这些简化使我们把精力集中于高层次的算法思想. 但应记住任何一个具体实现的成功都是与对这些隐含开销的控制分不开的.

### 6.1.3 几种分布式数据结构

在我们给出第一个基于分布式内存算法之前, 必须考虑数据分配问题. 如何将相关的矩阵和向量在网络上合理分配?

假定要将数据 $\boldsymbol{x} \in \mathbb{R}^n$ 分配到含有 $p$ 个处理器的网络上的各个本地内存. 设 $p$ 满足条件 $n = rp$, 解决此问题的两种 “标准” 方法分别是按行存储和按列存储.

在按列存储方法中, 将向量 $\boldsymbol{x}$ 看作 $r \times p$ 矩阵

$$x_{r\times p} = [x(1:r) \quad x(r+1:2r) \quad \cdots \quad x(1+(p-1)r:n)],$$

将每一列分配至一个处理器中, 即 $x(1+(\mu-1)r:\mu r) \in \text{Proc}(\mu)$. (在这里 $\in$ 表示 “存储于”.) 可以看出每个处理器拥有 $\boldsymbol{x}$ 的一段连续数据.

在按行存储方法中, 将向量 $\boldsymbol{x}$ 看作 $p \times r$ 矩阵

$$x_{p\times r} = [x(1:p) \quad x(p+1:2p) \quad \cdots \quad x((r-1)p+1:n)],$$

将每一行分配至一个处理器中, 即 $x(\mu:p:n) \in \text{Proc}(\mu)$. 按行存储有时被称作分配向量的轮流方法, 因为 $x$ 中的元素可被看作是一副扑克牌轮流发给每个处理器.

如果 $n$ 不是 $p$ 的整数倍, 则稍做改动后上述方案依然可行. 考虑 $n = 14, p = 4$ 的按列存储方法:

$$x^{\mathrm{T}} = [\underbrace{x_1\ x_2\ x_3\ x_4}_{\text{Proc(1)}} \mid \underbrace{x_5\ x_6\ x_7\ x_8}_{\text{Proc(2)}} \mid \underbrace{x_9\ x_{10}\ x_{11}}_{\text{Proc(3)}} \mid \underbrace{x_{12}\ x_{13}\ x_{14}}_{\text{Proc(4)}}].$$

一般地, 如果 $n = pr + q$ 且 $0 \leqslant q < p$, 则 $\text{Proc}(1), \cdots, \text{Proc}(q)$ 存储 $r+1$ 个元素, $\text{Proc}(q+1), \cdots, \text{Proc}(p)$ 存储 $r$ 个元素. 在按行存储方法中, 则只需令 $\text{Proc}(\mu)$ 包含元素 $x(\mu:p:n)$.

类似的选择同样适用于矩阵的数据分配. 如 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 且为简便起见令 $n = rp$, 则有下述四种分配的可能性.

| 方　向 | 方　式 | $\text{Proc}(\mu)$ 中元素 |
|---|---|---|
| 列 | 连续 | $A(:, 1+(\mu-1)r:\mu r)$ |
| 列 | 轮流 | $A(:, \mu:p:n)$ |
| 行 | 连续 | $A(1+(\mu-1)r:\mu r, :)$ |
| 行 | 轮流 | $A(\mu:p:n, :)$ |

分块矩阵也有类似的处理方案. 例如, 如果 $\boldsymbol{A} = [\boldsymbol{A}_1, \cdots, \boldsymbol{A}_N]$ 是按列分块的, 可令 $\text{Proc}(\mu)$ 来存储 $\boldsymbol{A}_i$, 其中 $i = \mu:p:N$.

### 6.1.4 环上的 gaxpy 运算

现在为 gaxpy 运算 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x} (\boldsymbol{A} \in \mathbb{R}^{n\times n}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^n)$ 给出一个环上的算法. 为清晰起见, 假定 $n = rp$, 其中 $p$ 为环中处理器数. 对 gaxpy 运算做如下划分:

$$\begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_p \end{bmatrix} + \begin{bmatrix} \boldsymbol{A}_{11} & \cdots & \boldsymbol{A}_{1p} \\ \vdots & & \vdots \\ \boldsymbol{A}_{p1} & \cdots & \boldsymbol{A}_{pp} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_p \end{bmatrix}, \tag{6.1.2}$$

其中 $\boldsymbol{A}_{ij} \in \mathbb{R}^{r\times r}$ 且 $\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{z}_i \in \mathbb{R}^r$. 假定计算之前 $\text{Proc}(\mu)$ 中已存储有 $\boldsymbol{x}_\mu, \boldsymbol{y}_\mu$ 和 $\boldsymbol{A}$ 的第 $\mu$ 个行块. 我们要求在计算结束时用 $\boldsymbol{z}_u$ 覆盖 $\boldsymbol{y}_\mu$. 从 $\text{Proc}(\mu)$ 的角度看, 运算

$$z_u = y_\mu + \sum_{\tau=1}^{p} A_{\mu\tau} x_\tau$$

涉及本地数据 $(A_{\mu\tau}, y_\mu, x_\mu)$ 和非本地数据 $(x_\tau, \tau \neq \mu)$. 为能得到 $x$ 的非本地部分，令它的子向量在环中循环. 以 $p=3$ 为例, 按如下所示对 $x_1, x_2, x_3$ 作循环:

| 步 | Proc(1) | Proc(2) | Proc(3) |
|---|---|---|---|
| 1 | $x_3$ | $x_1$ | $x_2$ |
| 2 | $x_2$ | $x_3$ | $x_1$ |
| 3 | $x_1$ | $x_2$ | $x_3$ |

每当 $x$ 的一个子向量到访时, 主处理器必须以合适的形式对其进行运算以产生新的和:

| 步 | Proc(1) | Proc(2) | Proc(3) |
|---|---|---|---|
| 1 | $y_1 = y_1 + A_{13}x_3$ | $y_2 = y_2 + A_{21}x_1$ | $y_3 = y_3 + A_{32}x_2$ |
| 2 | $y_1 = y_1 + A_{12}x_2$ | $y_2 = y_2 + A_{23}x_3$ | $y_3 = y_3 + A_{31}x_1$ |
| 3 | $y_1 = y_1 + A_{11}x_1$ | $y_2 = y_2 + A_{22}x_2$ | $y_3 = y_3 + A_{33}x_3$ |

概括地说, $x$ 的子向量走马灯似地流动, 共经过 $p$ 站. 处理器每接收到一个 $x$ 的子向量, 便进行一次 $r \times r$ 的 gaxpy 运算.

**算法 6.1.1**　假设 $A \in \mathbb{R}^{n\times n}, x \in \mathbb{R}^n, y \in \mathbb{R}^n$ 和 $z = y + Ax$. 如果含 $p$ 个处理器的环中的每个处理器执行如下的节点程序, 且 $n = rp$, 那么结束后 $z(1+(\mu-1)r:\mu r)$ 存于 Proc($\mu$) 的 $y_{\text{loc}}$ 变量中. 假定本地内存有如下初始化: $p, \mu$ (节点序号), left 和 right(左邻和右邻的序号), $n$, $\text{row} = 1+(\mu-1)r:\mu r, A_{\text{loc}} = A(\text{row},:), x_{\text{loc}} = x(\text{row}), y_{\text{loc}} = y(\text{row})$.

```
for t = 1 : p
    send (x_loc, right)
    recv (x_loc, left)
    τ = μ − t
    if τ ⩽ 0
        τ = τ + p
    end
    {x_loc = x(1 + (τ − 1)r : τr)}
    y_loc = y_loc + A_loc(:, 1 + (τ − 1)r : τr) x_loc
end
```

下标 $\tau$ 指明了当前所用的 $x$ 的子向量. 这步计算一旦完成, 就应对 $y$ 存储于本地的部分进行修正. **send-recv** 对将当前存储的 $x$ 的子向量发送到右邻并且等待从左邻接收下一组值. 因为只有在新的 $x$ 的子向量到达后才能开始对本地 $y$ 做修正, 这样便获得同步. 不可能出现任一处理器超前于其他处理器运算, 也不可能有 $x$ 的某子向量在走马灯流程中超过另一个子向量. 本算法专为环拓扑设计, 通信仅涉及

最近相邻的两个处理器. 此运算有完美的负载均衡, 这意味着每个处理器有完全相同的计算量和通信量. 非均衡负载将在 6.1.7 节中进一步讨论.

并行程序的设计涉及许多单处理器程序设计中碰不到的微妙的细节问题. 例如, 若我们不小心颠倒**send**和**recv**的次序, 那么每个处理器的节点程序执行的开始都在等待其左邻发送一个消息. 而这个左邻也在等待它的左邻发送消息, 便产生死锁.

### 6.1.5 通信成本

如果对发送和接收消息的开销模型化, 就可对通信成本做出估计. 为此假定涉及 $m$ 个浮点数的 **send** 和 **recv** 过程需要

$$\tau(m) = \alpha_d + \beta_d m \tag{6.1.3}$$

秒的执行时间. 这里 $\alpha_d$ 是 **send** 和 **recv** 过程初始化所需的时间, $\beta_d$ 是消息传送速率的倒数. 注意到, 此模型并没有考虑发送者与接收者之间的距离. 显然, 将一个消息在环上传送半圈要比仅在相邻处理器间传送耗时要多. 这就是为什么总要希望设计 (只要可能) 在相邻处理器间通信的分布式计算.

在算法 6.1.1 中的每一步都发送和接收一个 $r$ 维向量且执行了 $2r^2$ 次浮点运算. 如果每秒进行 $R$ 次浮点运算且 **recv** 过程中不存在等待时间, 那么每次 $y_{\text{loc}}$ 更新需大约 $(2r^2/R) + 2(\alpha_d + \beta_d r)$ 秒.

另一个指标是计算-通信比. 对于算法 6.1.1 可描述成

$$\frac{\text{计算时间}}{\text{通信时间}} \approx \frac{(2r^2/R)}{2(\alpha_d + \beta_d r)}.$$

这个比例量化了通信开销和计算开销间的比例关系. 显然, $r = n/p$ 增大时, 计算用时比也增加[①].

### 6.1.6 效率和加速比

一个 $p$ 个处理器的并行算法的效率由下式给出:

$$E = \frac{T(1)}{pT(p)},$$

其中 $T(k)$ 是在 $k$ 个处理器上执行此程序所需的时间. 如计算的速度是每秒 $R$ flop, 通信模型由 (6.1.3) 所给出, 则对算法 6.1.1 中的 $T(k)$ 的合理估计为

$$T(k) = \sum_{t=1}^{k} 2(n/k)^2/R + 2(\alpha_d + \beta_d(n/k)) = \frac{2n^2}{Rk} + 2\alpha_d k + 2\beta_d n \ ,$$

其中 $k > 1$. 我们假定没有等待时间. 如果 $k = 1$, 即无需通信, 则 $T(1) = 2n^2/R$. 可以看出效率

$$E = \frac{1}{1 + \dfrac{pR}{n}\left(\alpha_d \dfrac{p}{n} + \beta\right)}$$

① 我们指出, 这些简单度量对于节点能同时计算和通信的系统不能说明什么.

随 $n$ 增大而提高, 随 $p$ 或 $R$ 增大而降低. 在实际应用中, 只有用标准检测程序才是评估效率的唯一可靠方法.

与效率相关的一个概念是**加速比**. 如果

$$S = T_{\text{seq}}/T_{\text{par}},$$

则称解决一个特定问题的并行算法获得加速比 $S$, 其中 $T_{\text{par}}$ 是执行一个并行程序所需的时间, $T_{\text{seq}}$ 是单个处理器运行最有效的单处理程序时所需的时间. 对某些问题, 最快速的串行算法不是并行的, 因此, 加速比的评估涉及两种截然不同的算法.

### 6.1.7 均衡负载问题

如将算法 6.1.1 应用于下三角形矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 由于 (6.1.2) 中的 $A_{ij}$ 有一半为零, 则与 $y_{\text{loc}}$ 的更新相关的近一半浮点运算将没有必要进行了. 确切地说, 在第 $\mu$ 个处理器上, 满足 $\tau > \mu$ 的 $A_{\text{loc}}(:, 1+(\tau-1)r : \tau r)$ 为零. 因此, 如果保证 $y_{\text{loc}}$ 的更新按下述方法进行, 那么总的浮点运算次数将减半.

**if** $\tau \leqslant \mu$

$$y_{\text{loc}} = y_{\text{loc}} + A_{\text{loc}}(:, 1+(\tau-1)r : \tau r)x_{\text{loc}}$$

**end**

这样做解决了多余的运算问题但产生了不均衡负载问题. 在 $\text{Proc}(\mu)$ 上进行大约 $\mu r^2/2$ 次浮点运算, 它是关于处理器序号 $\mu$ 的增函数. 考虑下面 $r = p = 3$ 的例子:

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \hline z_4 \\ z_5 \\ z_6 \\ \hline z_7 \\ z_8 \\ z_9 \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha & \alpha & \alpha & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \beta & \beta & \beta & \beta & 0 & 0 & 0 & 0 & 0 \\ \beta & \beta & \beta & \beta & \beta & 0 & 0 & 0 & 0 \\ \beta & \beta & \beta & \beta & \beta & \beta & 0 & 0 & 0 \\ \hline \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & 0 & 0 \\ \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & 0 \\ \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma \end{array}\right] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \hline x_4 \\ x_5 \\ x_6 \\ \hline x_7 \\ x_8 \\ x_9 \end{bmatrix} + \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \hline y_4 \\ y_5 \\ y_6 \\ \hline y_7 \\ y_8 \\ y_9 \end{bmatrix}.$$

这里 $\text{Proc}(1)$ 处理 $\alpha$ 部分, $\text{Proc}(2)$ 处理 $\beta$ 部分, $\text{Proc}(3)$ 处理 $\gamma$ 部分.

然而, 如果处理器 1, 2 和 3 分别计算 $(z_1, z_4, z_7), (z_2, z_5, z_8)$ 和 $(z_3, z_6, z_9)$, 就会导致近似的均衡负载:

$$\begin{bmatrix} z_1 \\ z_4 \\ z_7 \\ \hline z_2 \\ z_5 \\ z_8 \\ \hline z_3 \\ z_6 \\ z_9 \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta & \beta & \beta & \beta & 0 & 0 & 0 & 0 & 0 \\ \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & 0 & 0 \\ \hline \alpha & \alpha & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta & \beta & \beta & \beta & \beta & 0 & 0 & 0 & 0 \\ \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & 0 \\ \hline \alpha & \alpha & \alpha & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta & \beta & \beta & \beta & \beta & \beta & 0 & 0 & 0 \\ \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma & \gamma \end{array}\right] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \hline x_4 \\ x_5 \\ x_6 \\ \hline x_7 \\ x_8 \\ x_9 \end{bmatrix} + \begin{bmatrix} y_1 \\ y_4 \\ y_7 \\ \hline y_2 \\ y_5 \\ y_8 \\ \hline y_3 \\ y_6 \\ y_9 \end{bmatrix}.$$

运算量虽仍随 $\mu$ 增大而增大, 但在 $n \gg p$ 时不明显.

设计一般的算法需要一些对下标的操作. 假定在 Proc($\mu$) 赋初值 $\boldsymbol{A}_{\text{loc}} = A(\mu : p : n, :)$ 和 $y_{\text{loc}} = y(\mu : p : n)$, 并假定连续的 $\boldsymbol{x}$ 子向量如前述一样循环. 如在某步 $x_{\text{loc}}$ 包含 $x(1+(\tau-1)r : \tau r)$, 则更新

$$y_{\text{loc}} = y_{\text{loc}} + A_{\text{loc}}(:, 1+(\tau-1)r : \tau r)x_{\text{loc}}$$

的实施为

$$y(\mu : p : n) = y(\mu : p : n) + A(\mu : p : n, 1+(\tau-1)r : \tau r)x(1+(\tau-1)r : \tau r).$$

为在 $y_{\text{loc}}$ 的计算中利用 $\boldsymbol{A}$ 的三角结构, 我们用双重循环形式表示 gaxpy 运算:

**for** $\alpha = 1 : r$

  **for** $\beta = 1 : r$

    $y_{\text{loc}}(\alpha) = y_{\text{loc}}(\alpha) + A_{\text{loc}}(\alpha, \beta + (\tau-1)r)x_{\text{loc}}(\beta)$

  **end**

**end**

$\boldsymbol{A}_{\text{loc}}$ 代表 $A(\mu+(\alpha-1)p, \beta+(\tau-1)r)$, 它当列下标大于行下标时为零. 基于这种思想, 我们缩短内循环的范围, 得到下列算法.

**算法 6.1.2** 假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 和 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$. 设 $n = rp$ 且 $\boldsymbol{A}$ 为下三角形矩阵. 如果含 $p$ 个处理器的环中的每个处理器执行如下的节点程序, 那么结束时 $z(\mu : p : n)$ 存于 Proc($\mu$) 的 $y_{\text{loc}}$ 变量中. 假定本地内存有如下初始值: $p, \mu$ (节点序号), left 和 right(左邻和右邻标识号), $n, A_{\text{loc}} = A(\mu : p : n, :), y_{\text{loc}} = y(\mu : p : n)$ 和 $x_{\text{loc}} = x(1+(\mu-1)r : \mu r)$.

$r = n/p$

**for** $t = 1 : p$

  **send** ($x_{\text{loc}}$, right)

  **recv** ($x_{\text{loc}}$, left)

```
        τ = μ − t
        if τ ⩽ 0
            τ = τ + p
        end
        {x_loc = x(1 + (τ − 1)r : τr)}
        for α = 1 : r
            for β = 1 : μ + (α − 1)p − (τ − 1)r
                y_loc(α) = y_loc(α) + A_loc(α, β + (τ − 1)r)x_loc(β)
            end
        end
    end
```

下标值不得不在节点空间和全局空间之间来回改变是分布式矩阵计算中需要小心谨慎和 (希望) 得到编译程序支持的一个方面.

### 6.1.8　平衡协调问题 (Tradeoffs)

类似于 1.1 节, 我们给出一个基于列的 gaxpy 运算并预测它的性能. 对 $\boldsymbol{A}$ 作列分块:

$$\boldsymbol{A} = [\boldsymbol{A}_1, \cdots, \boldsymbol{A}_p], \quad \boldsymbol{A}_i \in \mathbb{R}^{n\times r}, \quad r = n/p,$$

gaxpy 运算 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$ 变成

$$\boldsymbol{z} = \boldsymbol{y} + \sum_{\mu=1}^{p} \boldsymbol{A}_\mu \boldsymbol{x}_\mu,$$

其中 $\boldsymbol{x}_\mu = x(1 + (\mu - 1) : \mu r)$. 假定 Proc($\mu$) 中存储了 $\boldsymbol{A}_\mu$ 和 $\boldsymbol{x}_\mu$. 它对整个 gaxpy 的贡献是提供了积 $\boldsymbol{A}_\mu \boldsymbol{x}_\mu$ 而且只涉及本地数据. 然而, 还需对这些积做累加. 让 Proc(1) 来承担累加的工作, 假定它已存储 $\boldsymbol{y}$. 因此算法的思想是每个处理器都计算 $\boldsymbol{A}_\mu x_\mu$ 然后将结果送至 Proc(1).

**算法 6.1.3**　假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 和 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$, 如果含 $p$ 个处理器的环中的每个处理器执行如下的节点程序, 且 $n = rp$, 则结束后, $\boldsymbol{z}$ 存于 Proc(1) 中. 假定本地内存有如下初始值: $p, \mu$ (节点序号), $n, x_{\text{loc}} = x(1 + (\mu - 1)r : \mu r), A_{\text{loc}} = A(:,1 + (\mu - 1)r : \mu r)$, 和 (仅在 Proc(1))$y_{\text{loc}} = y$.

```
if μ = 1
    y_loc = y_loc + A_loc x_loc
    for t = 2 : p
        recv (ω_loc, t)
        y_loc = y_loc + ω_loc
    end
else
    ω_loc = A_loc x_loc
```

**send** $(\omega_{\text{loc}}, 1)$

**end**

初看起来此算法与基于行的算法 6.1.1 相比似乎缺少吸引力. Proc(1) 的额外工作意味着它需承担原来的

$$\frac{2n^2/p + np}{2n^2/p} = 1 + \frac{p^2}{2n}$$

倍的工作量, 并且处理原来的 $p$ 倍的消息. 这种不均衡在 $n \gg p$ 和通信参数 $\alpha_d$ 和 $\beta_d$ 足够小时显得无足轻重. 另一个可能优势在于算法 6.1.3 对长度为 $n$ 的向量进行运算而算法 6.1.1 对长度为 $n/p$ 的向量进行运算. 在节点允许的条件下, 加长向量会提高性能.

算法 6.1.1 和算法 6.1.3 这一简单比较再次提醒我们, 同样的计算采用不同的实现方法能有很不同运算特性.

### 6.1.9 共享内存系统

接下来讨论基于共享内存的多处理机上的 gaxpy 问题. 在此环境中每个处理器去访问公共的全局内存, 如图 6.1.2 所示. 通过不断读写位于全局内存的*全局变量*来实现处理器间的通信. 每个处理器拥有自己的*本地内存*并执行*本地程序*, 在执行过程中, 数据不断流入流出全局内存.

共享内存计算中所考虑的所有问题都以不同的形式摆在我们面前. 整个程序应该是*均衡负载的*, 对计算过程的安排应使任一独立的处理器等待它计算所用数据时间尽可能少. 全局内存和本地内存之间的数据交换必须认真处理, 因为数据交换在整个系统开销中占相当比重. (它类似于分布式内存系统中的处理器间通信和如 1.4.5 节中所述的不同级的存储之间数据的流动). 处理器与共享内存之间的物理连接的方式非常重要, 它影响到算法设计. 然而, 为简化问题, 我们将系统的这一方面看作如图 6.1.2 所示的黑盒子.

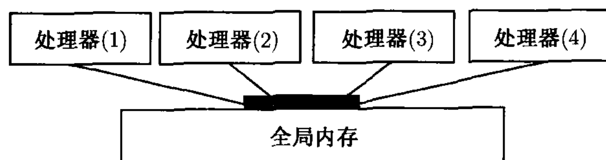



图 6.1.2 四处理器共享内存系统

### 6.1.10 共享内存的 gaxpy 算法

考虑将 $n \times n$ 的 gaxpy 问题 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$ 做如下划分

$$\begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_p \end{bmatrix} + \begin{bmatrix} \boldsymbol{A}_1 \\ \vdots \\ \boldsymbol{A}_p \end{bmatrix} \boldsymbol{x}. \tag{6.1.4}$$

在此假定 $n = rp$ 且 $\boldsymbol{A}_\mu \in \mathbb{R}^{r\times n}, \boldsymbol{y}_\mu, \boldsymbol{z}_\mu \in \mathbb{R}^r$. 我们用以下算法来介绍基本思想和记号.

**算法 6.1.4**　*假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 存储于全局内存中且能被 $p$ 个处理器访问. 如果 $n = rp$, 每一个处理器执行如下的节点程序, 那么结束时 $\boldsymbol{y}$ 被 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$ 覆盖. 假定每个本地内存有如下初始值: $p, \mu$ (节点序号), $n$.*

$r = n/p$

$\text{row} = 1 + (\mu - 1)r : \mu r$

$x_{\text{loc}} = x$

$y_{\text{loc}} = y(\text{row})$

**for** $j = 1 : n$

　　$a_{\text{loc}} = A(\text{row}, j)$

　　$y_{\text{loc}} = y_{\text{loc}} + a_{\text{loc}} x_{\text{loc}}(j)$

**end**

$y(\text{row}) = y_{\text{loc}}$

假定在每个处理器中都有一份这个程序. 属于单个处理器的本地浮点变量带有 "loc" 下标.

在算法 6.1.4 的执行中, 数据不断流入流出全局内存. 在循环前有两次读全局内存操作 ($x_{\text{loc}} = x$ 和 $y_{\text{loc}} = y(\text{row})$), 循环中有一次读操作 ($a_{\text{loc}} = A(\text{row}, j)$), 循环后有一次写操作 ($y(\text{row}) = y_{\text{loc}}$).

对于 $y$ 在全局内存中的某一部分, 只有一个处理器对其进行写操作, 因此不必对参加计算的处理器进行同步. 在整个 gaxpy 运算过程中, 每个处理器都有完全独立的工作, 因此不必监视其他处理器的工作情况. 因为对工作的分配是在执行前已决定好的, 这种计算称为**静态调度**.

如果 $\boldsymbol{A}$ 是下三角形矩阵, 则在算法 6.1.4 中要采取措施以保证负载均衡. 正如 6.1.7 节中所讨论的, 轮流分配是一种有效办法. 将 $z(\mu : p : n) = y(\mu : p : n) + A(\mu : p : n, :)x$ 分配给 $\text{Proc}(\mu)$ 就将 $n^2$ 次浮点运算在 $p$ 个处理器间有效地分配.

### 6.1.11　内存间数据通信开销

重要的是要认识到整个算法的表现很强地依赖于对全局内存的读写开销. 如果一次数据传送中涉及 $m$ 个浮点数, 则传送时间之模型为

$$\tau(m) = \alpha_s + \beta_s m. \tag{6.1.5}$$

参数 $\alpha_s$ 代表通信过程启动开销, $\beta_s$ 是传送速度的倒数. 在分布式环境中我们是用相同的方式建立处理器间的通信模型 (见 (6.1.3)).

数一下算法 6.1.4 中共享内存的所有读写, 我们发现每个处理器在与全局内存通信时花费的时间为

$$T \approx (n+3)\alpha_s + \frac{n^2}{p}\beta_s.$$

我们组织计算的方式是使每次从共享内存中读取 $A(\text{row},:)$ 的一列. 如果本地内存足够大, 算法 6.1.4 中的循环可改写为

$$A_{\text{loc}} = A(\text{row},:)$$
$$y_{\text{loc}} = y_{\text{loc}} + A_{\text{loc}}x_{\text{loc}}.$$

这使通信开销变为

$$\tilde{T} \approx 3\alpha_s + \frac{n^2}{p}\beta_s,$$

当启动时间 $\alpha_s$ 很大时, 是显著的改进.

### 6.1.12 障碍同步

下面考虑面向列的 gaxpy 运算 (算法 6.1.4) 的共享内存形式. 假定 $n = rp$ 和 $\text{col} = 1 + (\mu - 1)r : \mu r$. 一个合理的想法是用一个全局数组 $W(1:n, 1:p)$ 来存放每个处理器产生的积 $A(:,\text{col})x(\text{col})$, 然后选定一个处理器 (如 Proc(1)) 来做累加.

$A_{\text{loc}} = A(:,\text{col}); x_{\text{loc}} = x(\text{col}); \omega_{\text{loc}} = A_{\text{loc}}x_{\text{loc}}; W(:,\mu) = \omega_{\text{loc}}$
**if** $\mu = 1$
  $y_{\text{loc}} = y$
  **for** $j = 1:p$
    $\omega_{\text{loc}} = W(:,j)$
    $y_{\text{loc}} = y_{\text{loc}} + \omega_{\text{loc}}$
  **end**
  $y = y_{\text{loc}}$
**end**

然而, 这个方案有一个严重缺陷, 它不能保证 Proc(1) 在做累加前 $W(1:n, 1:p)$ 已被赋值.

我们需要的是有一种同步机制将 Proc(1) 的累加运算延迟至所有的处理器都计算完各自的结果并存储到 $W$ 数组之后才开始. 为此, 许多共享内存系统都支持一种阻塞机制**barrier**, 它在下列算法中引入.

**算法 6.1.5** *假设 $A \in \mathbb{R}^{n\times n}, x, y \in \mathbb{R}^n$ 存储于全局内存中且能被 $p$ 个处理器访问. 如果 $n = rp$, 每一个处理器执行如下的节点程序, 那么结束时 $y$ 被 $y + Ax$ 覆盖. 假定本地内存有如下初始值: $p, \mu$ (节点序号), $n$.*

$r = n/p; \text{col} = 1 + (\mu - 1)r : \mu r; A_{\text{loc}} = A(:,\text{col}); x_{\text{loc}} = x(\text{col})$
$\omega_{\text{loc}} = A_{\text{loc}}x_{\text{loc}}$
$W(:,\mu) = \omega_{\text{loc}}$
**barrier**
**if** $\mu = 1$
  $y_{\text{loc}} = y$
  **for** $j = 1:p$

$\omega_{\text{loc}} = W(:, j)$

$y_{\text{loc}} = y_{\text{loc}} + \omega_{\text{loc}}$

**end**

$y = y_{\text{loc}}$

**end**

为了理解**barrier**, 可将一台处理器看作或是截住的或是自由的. 当执行**barrier**语句时, 处理器被截住, 程序被暂停. 当 $p$ 个处理器都被截住时, 所有处理器都返回到自由状态并恢复执行程序. 可将**barrier**想象为 $p$ 个处理器想横渡一条急流. 为了安全, 他们在跨越前都到岸边集合. 当最后一个抵达后, 他们一起涉过急流, 然后分别继续各自的旅行.

在算法 6.1.5 中, 处理器算完矩阵乘向量的积后被截住. 我们并不能预见处理器被截住的顺序. 但一旦最后一个处理器也被截住, 他们全被解放, Proc(1) 就能开始执行向量累加运算.

### 6.1.13 动态调度

如不采取选定某一个处理器来做向量累加的方法, 我们可以让每个处理器将其计算结果直接加到全局变量 $y$ 中. 对 Proc($\mu$) 来说, 这意味着要执行如下指令:

$r = n/p; \text{col} = 1 + (\mu - 1)r : \mu r; A_{\text{loc}} = A(:, \text{col}); x_{\text{loc}} = x(\text{col})$

$\omega_{\text{loc}} = A_{\text{loc}} x_{\text{loc}};$

$y_{\text{loc}} = y; y_{\text{loc}} = y_{\text{loc}} + \omega_{\text{loc}}; y = y_{\text{loc}}$

这样, 一个问题包含读–更新–写三部曲:

$$y_{\text{loc}} = y; \quad y_{\text{loc}} = y_{\text{loc}} + \omega_{\text{loc}}; \quad y = y_{\text{loc}}$$

事实上, 如果不只一个处理器同时执行这个代码段, 则可能会造成信息丢失. 考虑如下执行顺序

Proc(1) 读 $\boldsymbol{y}$

Proc(2) 读 $\boldsymbol{y}$

Proc(1) 写 $\boldsymbol{y}$

Proc(2) 写 $\boldsymbol{y}$

由于 Proc(1) 和 Proc(2) 对同一个 $\boldsymbol{y}$ 进行操作, Proc(1) 计算的结果没有起到应起的作用. 由于 Proc(2) 的写操作使得 Proc(1) 的写操作无效.

为防止这种情况发生, 大多数共享内存机器支持*临界区*(critical section) 的概念. 临界区是节点程序中特殊的孤立段, 它需要有 "钥匙" 才能进入. 在整个机器中, 由于只有一把钥匙, 因此在任一特定的时刻, 只有一个处理器能执行临界区程序.

**算法 6.1.6** *假设 $\boldsymbol{A} \in \mathbb{R}^{n \times n}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ 存储于全局内存中且能被 $p$ 个处理器访问. 如果 $n = rp$, 每个处理器执行如下的节点程序, 那么在结束时, $\boldsymbol{y}$ 被 $\boldsymbol{y} + \boldsymbol{A}\boldsymbol{x}$*

*覆盖*. 假定本地内存有如下初始值: $p, \mu$ (节点序号), $n$.

$r = n/p; \text{col} = 1 + (\mu - 1)r : \mu r; A_{\text{loc}} = A(:, \text{col}); x_{\text{loc}} = x(\text{col})$

$\omega_{\text{loc}} = A_{\text{loc}} x_{\text{loc}}$

**begin critical section**

$y_{\text{loc}} = y$

$y_{\text{loc}} = y_{\text{loc}} + \omega_{\text{loc}}$

$y = y_{\text{loc}}$

**end critical section**

临界区概念的引入保证了 $y$ 能够正确地被更新. 由于此算法中累加的顺序是在计算进行过程中决定的, 因此称为*动态调度*. 在解决非规则问题时动态调度是非常重要的.

## 习 题

**6.1.1** 修改算法 6.1.1 使其能处理任意的 $n$ 值.

**6.1.2** 修改算法 6.1.2 使其能有效处理上三角形矩阵.

**6.1.3** (a) 修改算法 6.1.3 和算法 6.1.4 使得对每个处理器, 给定正数 $m$, 可以用 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}^m \boldsymbol{x}$ 来覆盖 $\boldsymbol{y}$. (b) 修改算法 6.1.3 和算法 6.1.4 用 $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{A}^{\mathrm{T}} \boldsymbol{x}$ 覆盖 $\boldsymbol{y}$.

**6.1.4** 修改算法 6.1.3 使算法执行完后 $\boldsymbol{A} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}$ 的第 $\mu$ 个列块存储于 Proc($\mu$) 的本地数组 $A_{\text{loc}}$ 中.

**6.1.5** 修改算法 6.1.4 使得 (a)$\boldsymbol{A}$ 被外积更新 $\boldsymbol{A} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}$ 所覆盖. (b) 用 $\boldsymbol{A}^2 \boldsymbol{x}$ 覆盖掉 $\boldsymbol{x}$, (c) $\boldsymbol{y}$ 被 $\boldsymbol{y} + \boldsymbol{A}^k \boldsymbol{x}$ 方向上的单位 2 范数向量覆盖. (d) 当 $\boldsymbol{A}$ 为下三角形矩阵时, 该算法能有效地执行.

## 本节注释与参考文献

下列关于并行计算的专著都有专门几章讨论矩阵计算:


G. C. Fox, M. A. Johnson, G. A. Lyzenga, S. W. Otto, J. K. Salmon and D. W. Walker(1988). *Solving Problems on Concurrent Processors, Volume* 1. Prentice Hall, Englewood Cliffs, NJ.

D. P. Bertsekas and J. N. Tsitsiklis(1989). *Parallel and Distributed Computation: Numerical Methods*, Prentice Hall, Englewood Cliffs, NJ.

S. Lakshmivarahan and S. K. Dhall(1990). *Analysis and Design of Paraller Algorithms: Arithmetic and Matrix Problems*, McGraw-Hill, New York.

T. L. Freeman and C. Phillips(1992). *Parallel Numerical Algorithms*, Prentice Hall. New York.

F. T. Leighton(1992). *Introduction to Parallel Algorithms and Architectures*, Morgan Kaufmann, San Mateo, CA.

G. C. Fox, R. D. Williams, and P. C. Messina(1994). *Parallel Computing Works!*, Morgan Kaufmann, San Francisco.

V. Kumar, A. Grama, A. Gupta and G. Kaypis(1994). *Introduction to Parallel Computing: Design and Analysis of Algorithms*, Benjamin/Cummings, Reading, MA.

E. F. Van de Velde(1994). *Concurrent Scientific Computing*, Springer-Verlag, New York.

M. Cosnard and D. Trystram(1995). *Parallel Algorithms and Architectures*, International Thomson Computer Press, New York.

以下是专门论述并行矩阵计算的一些文章：

V. Fadeeva and D. Fadeev(1977). "Parallel Computations in Linear Algebra," *Kibernetica* 6, 28–40.

D. Heller(1978). "A Survey of Parallel Algorithms in Numerical Linear Algebra," *SIAM Review* 20, 740–777.

J. M. Ortega and R. G. Voigt(1985). "Solution of Partial Differential Equations on Vector and Parallel Computers," *SIAM Review* 27, 149–240.

J. J. Dongarra and D. C. Sorensen(1986). "Linear Algebra on High Performance Computers," *Appl. Math. and Comp.* 20, 57–88.

K. A. Gallivan, R. J. Plemmons, and A. H. Sameh(1990). "Parallel Algorithms for Dense Linear Algebra Computations," *SIAM Review* 32, 54–135.

J. W. Demmel, M. T. Heath, and H. A. van der Vorst(1993). "Parallel Numerical Linear Algebra," in *Acta Numerica* 1993, Cambridge University Press.

此外还有：

B. N. Datta(1989). "Parallel and Large-Scale Matrix Computations in Control: Some Ideas," *Lin. Alg. and Its Applic.* 121. 243–264.

A. Edelman(1993). "Large Dense Numerical Linear Algebra in 1993: The Parallel Computing Influence," *Int'l J. Supercomputer Appl.* 7, 113–128.

在分布式内存环境中，管理和设计通信工作非常重要，亦很困难. 见:

L. Adams and T. Crockett(1984). "Modeling Algorthm Execution Time on Processor Arrays," *Computer* 17, 38–43.

D. Gannon and J. Van Rosendale(1984). "On the Impact of Communication Complexity on the Design of Parallel Numerical Algorithms," *IEEE Trans. Comp.* C-33, 1180–1194.

S. L. Johnsson(1987). "Communication Efficient Basic Linear Algebra Computations on Hypercube Multiprocessors," *J. Parallel and Distributed Computing*, No. 4, 133–172.

Y. Saad and M. Schultz (1989). "Data Communication in Hypercubes," *J. Dist. Parallel Comp.* 6, 115–135.

Y. Saad and M. H. Schultz(1989). "Data Communication in Parallel Architectures," *J. Dist. Parallel Comp.* 11, 131–150.

想快速了解分布式内存系统上基本线性代数计算，请参见：

O. McBryan and E. F. van de Velde(1987). "Hypercube Algorithms and Implementations," *SIAM J. Sci. and Stat. Comp.* 8, s277–s287.

S. L. Johnsson and C. T. Ho(1988). "Matrix Transposition on Boolean $n$-cube Configured

Ensemble Architectures," *SIAM J. Matrix Anal. Appl.* 9, 419–454.

T. Dehn, M. Eiermann, K. Giebermann, and V. Sperling(1995). "Structured Sparse Matrix Vector Multiplication on Massively Parallel SIMD Architectures," *Parallel Computing* 21, 1867–1894.

J. Choi, J. J. Dongarra, and D. W. Walker(1995). "Parallel Matrix Transpose Algorithms on Distributed Memory Concurrent Computers," *Parallel Computing* 21, 1387–1406.

L. Colombet, Ph. Michallon, and D. Trystram (1996). "Parallel Matrix-Vector Product on Rings with a Minimum of Communication," *Parallel Computing* 22, 289–310.

并行算法的实现是一项很有挑战性的工作, 利用编译器和一些相关工具来处理细节是很重要的, 参阅:

D. P. O'Leary and G. W. Stewart(1986). "Assignment and Scheduling in Parallel Matrix Factorization," *Lin. Alg. and Its Applic.* 77, 275–300.

J. Dongarra and D. C. Sorensen(1987). "A Portable Environment for Developing Parallel Programs", *Parallel Computing* 5, 175–186.

K. Connolly, J. J. Dongraar, D. Sorensen, and J. Patterson(1988). "Programming Methodology and Performance Issues for Advanced Computer Architectures," *Parallel Computing* 5, 41–58.

P. Jacobson, B. Kagstrom, and M. Rannar(1992). "Algorithm Development for Distributed Memory Multicomputers Using Conlab," *Scientific Programming*, 1, 185–203.

C. Ancourt, F. Coelho, F. Irigoin, and R. Keryell(1993). "A Linear Algebra Framework for Static HPF Code Distribution," *Proceedings of the 4th Workshop on Compilers for Parallel Computers*, Delft, The Netherlands.

D. Bau, I. Kodukula, V. Kotlyar, K. Pinali, and P. Stodghill (1993). "Solving Alignment Using Elementary Linear Algebra," in *Proceedings of the 7th International Workshop on Languages and Compilers for Parallel Computing*, Lecture Notes in Computer Science 892. Springer-Verlag, New York, 46–60.

M. Wolfe(1996). *High Performance Compilers for Parallel Computers*, Addison-Wesley, Reading MA.

## 6.2 矩阵乘法

在本节中我们给出矩阵与矩阵相乘的两个并行算法. 共享内存的算法用来说明分块和均衡负载的效果. 基于环面结构的算法表达了二维数据流动思想.

### 6.2.1 分块 gaxpy 算法

假设 $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{n\times n}$, $\boldsymbol{B}$ 是上三角形矩阵, 考虑在含 $p$ 个处理器的共享内存系统中计算矩阵乘法修正:

$$\boldsymbol{D}=\boldsymbol{C}+\boldsymbol{A}\boldsymbol{B}. \tag{6.2.1}$$

假定 $n=rkp$, 且将上式分块:

$$[\boldsymbol{D}_1, \cdots, \boldsymbol{D}_{k\cdot p}] = [\boldsymbol{C}_1, \cdots, \boldsymbol{C}_{k\cdot p}] + [\boldsymbol{A}_1, \cdots, \boldsymbol{A}_{k\cdot p}][\boldsymbol{B}_1, \cdots, \boldsymbol{B}_{k\cdot p}], \tag{6.2.2}$$

其中每个列块的宽度为 $r = n/(kp)$. 如果

$$\boldsymbol{B}_j = \begin{bmatrix} \boldsymbol{B}_{1j} \\ \vdots \\ \boldsymbol{B}_{jj} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \boldsymbol{B}_{ij} \in \mathbb{R}^{r\times r},$$

则

$$\boldsymbol{D}_j = \boldsymbol{C}_j + \boldsymbol{A}\boldsymbol{B}_j = \boldsymbol{C}_j + \sum_{\tau=1}^{j} \boldsymbol{A}_\tau \boldsymbol{B}_{\tau j}. \tag{6.2.3}$$

计算 $\boldsymbol{D}_j$ 所需的 flop 数为

$$f_j = 2nr^2 j = \left(\frac{2n^3}{k^2p^2}\right) j.$$

因 $\boldsymbol{B}$ 是上三角形矩阵, $f_j$ 是关于 $j$ 的增函数. 正如我们在前一节所发现的, 由三角形矩阵结构带来的负载不均衡问题可由轮流分配法来解决. 这意味着分配给 $\text{Proc}(\mu)$ 的任务是计算满足 $j = \mu : p : kp$ 的 $\boldsymbol{D}_j$.

**算法 6.2.1** **假设 $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ 是 $n\times n$ 矩阵, 存储于全局内存中且能被 $p$ 个处理器访问. 如果 $\boldsymbol{B}$ 是上三角形矩阵且 $n = rkp$, 每个处理器执行如下的节点程序, 那么结束后 $\boldsymbol{C}$ 被 $\boldsymbol{D} = \boldsymbol{C} + \boldsymbol{A}\boldsymbol{B}$ 覆盖. 假定本地内存有如下初始值: $n, r, k, p, \mu$ (节点序号).**

```
for j = μ : p : kp
    {计算 D_j.}
    B_loc = B(1 : jr, 1 + (j − 1)r : jr)
    C_loc = C(:, 1 + (j − 1)r : jr)
    for τ = 1 : j
        col = 1 + (τ − 1)r : τr
        A_loc = A(:, col)
        C_loc = C_loc + A_loc B_loc(col, :)
    end
    C(:, 1 + (j − 1)r : jr) = C_loc
end
```

我们用一个关于参数 $k$ 的函数来评估负载的均衡度. Proc $(\mu)$ 所需的 flop 数为

$$F(\mu) = \sum_{i=1}^{k} f_{\mu+(i-1)p} \approx \left(k\mu + \frac{k^2p}{2}\right)\frac{2n^3}{k^2p^2}.$$

从 flop 数来看, 商 $F(p)/F(1)$ 是负载平衡的一个量度. 由

$$\frac{F(p)}{F(1)}=\frac{kp+k^2p/2}{k+k^2p/2}=1+\frac{2(p-1)}{2+kp}$$

可以看出随 $k$ 增大计算均衡度增加. 做类似的分析可看出通信开销也随 $k$ 增加而趋于平衡.

另一方面, 算法 6.2.1 中对全局内存的读写次数随 $k$ 的平方增大而增加. 如 (6.1.5) 中启动参数 $\alpha_s$ 很大, 那么性能随 $k$ 增加而降低.

由于这两种相矛盾的影响, $k$ 的最佳选择是与机器相关的. 如果通信加快了, 则可把任务分得很小而不带来困难, 因而就容易获得负载平衡. 具有这种性质的多处理机支持用单位被分得很小的并行算法. 然而, 如果在具有高性能节点的系统中单位划分过小, 则节点程序就会因没有足够的本地线性代数而不能用二级或三级运算. 再次强调, 标准检测程序是用来评估的唯一方式.

### 6.2.2 基于环面存储结构 (Torus) 的运算

环面 (Torus) 是行和列都为环的二维处理器数组, 参见图 6.2.1. 在这里处理器序号是一个有序对, 且每个处理器有四个邻居. 在如下示例中, Proc(1, 3) 的西邻是 Proc(1, 2), 东邻是 Proc(1, 4), 南邻是 Proc(2, 3), 北邻是 Proc(4, 3).

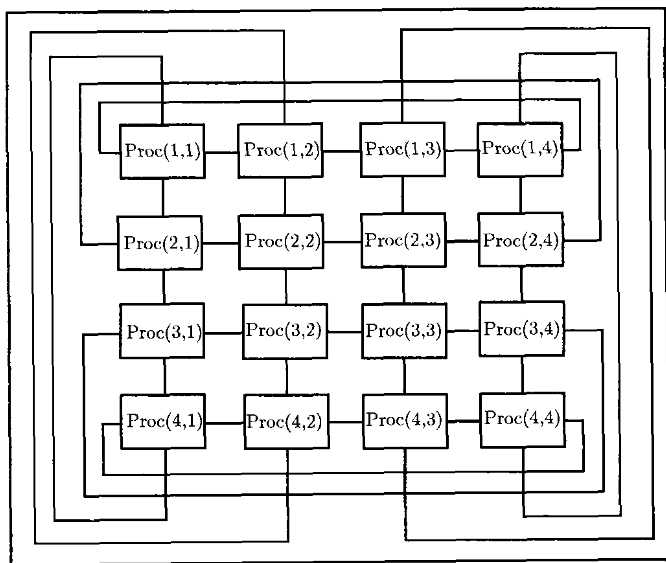



图 6.2.1 $4\times4$ 的环面

为说明如何组织环面结构上的矩阵计算, 我们设计一个算法来计算矩阵乘法 $\boldsymbol{D}=\boldsymbol{C}+\boldsymbol{AB}$, 其中 $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{n\times n}$. 假设环面结构是 $p_1\times p_1$ 的, 且有 $n=rp_1$. 将 $\boldsymbol{A}=(\boldsymbol{A}_{ij}),\boldsymbol{B}=(\boldsymbol{B}_{ij}),\boldsymbol{C}=(\boldsymbol{C}_{ij})$ 看作是含有 $p_1\times p_1$ 个块, 每块大小为 $r\times r$ 的分块矩阵. 假设 Proc$(i,j)$ 储存了 $\boldsymbol{A}_{ij},\boldsymbol{B}_{ij},\boldsymbol{C}_{ij}$, 它的任务是用

$$D_{ij} = C_{ij} + \sum_{k=1}^{p_1} A_{ik} B_{kj}$$

来覆盖 $C_{ij}$. 我们从 $p_1 = 3$ 的情形来导出一般算法, 将环面存储结构以如下形式表示:

| Proc(1, 1) | Proc(1, 2) | Proc(1, 3) |
|---|---|---|
| Proc(2, 1) | Proc(2, 2) | Proc(2, 3) |
| Proc(3, 1) | Proc(3, 2) | Proc(3, 3) |

我们集中注意力于 Proc(1, 1) 以及计算

$$D_{11} = C_{11} + A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31}.$$

假定用于确定分块点积的六个输入量在环面结构的存储位置如下:

| | | | | | |
|---|---|---|---|---|---|
| $A_{11}$ | $B_{11}$ | $A_{12}$ | • | $A_{13}$ | • |
| • | $B_{21}$ | • | • | • | • |
| • | $B_{31}$ | • | • | • | • |

(表格中的点不必理会, 它们在后面被 $A_{ij}$ 和 $B_{ij}$ 所代替).

我们让 $A$ 的第一行块和 $B$ 的第一列块都经过 Proc(1, 1) 依次轮转. 这样 $A_{11}$ 和 $B_{11}, A_{12}$ 和 $B_{21}, A_{13}$ 和 $B_{31}$, 将成对相遇、相乘, 然后与 $C_{\text{loc}}$ 相加.

| | | | | | |
|---|---|---|---|---|---|
| $A_{12}$ | $B_{21}$ | $A_{13}$ | • | $A_{11}$ | • |
| • | $B_{31}$ | • | • | • | • |
| • | $B_{11}$ | • | • | • | • |

$$C_{\text{loc}} = C_{\text{loc}} + A_{12}B_{21}$$

| | | | | | |
|---|---|---|---|---|---|
| $A_{13}$ | $B_{31}$ | $A_{11}$ | • | $A_{12}$ | • |
| • | $B_{11}$ | • | • | • | • |
| • | $B_{21}$ | • | • | • | • |

$$C_{\text{loc}} = C_{\text{loc}} + A_{13}B_{31}$$

| | | | | | |
|---|---|---|---|---|---|
| $A_{11}$ | $B_{11}$ | $A_{12}$ | • | $A_{13}$ | • |
| • | $B_{21}$ | • | • | • | • |
| • | $B_{31}$ | • | • | • | • |

$$C_{\text{loc}} = C_{\text{loc}} + A_{11}B_{11}$$

做完上述三步后, Proc(1, 1) 的本地数组中就有了 $D_{11}$.

在环面存储结构中我们安排的数据流动是 $A_{1j}$ 向西流动, $B_{i1}$ 向北流动. 显然 Proc(1, 1) 应执行如下节点程序:

**for** $t = 1:3$
　　**send** $(A_{\text{loc}}, \text{west})$
　　**send** $(B_{\text{loc}}, \text{north})$
　　**recv** $(A_{\text{loc}}, \text{east})$
　　**recv** $(B_{\text{loc}}, \text{south})$
　　$C_{\text{loc}} = C_{\text{loc}} + A_{\text{loc}}B_{\text{loc}}$
**end**

下述的 **send-recv-send-recv** 的顺序:

```
for t = 1 : 3
    send (A_loc, west)
    recv (A_loc, east)
    send (B_loc, north)
    recv (B_loc, south)
    C_loc = C_loc + A_loc B_loc
end
```

也是可行的. 但是由于 $\boldsymbol{B}$ 的子块必须等到新的 $\boldsymbol{A}$ 的子块到达后才能发送, 这就带来了不必要的延迟.

下面讨论 Proc(1, 2), Proc(1, 3), Proc(2, 1) 和 Proc(3, 1) 的工作. 仅就目前的情形, 这些处理器的作用只是帮着循环块 $\boldsymbol{A}_{11}, \boldsymbol{A}_{12}, \boldsymbol{A}_{13}$ 和 $\boldsymbol{B}_{11}, \boldsymbol{B}_{21}, \boldsymbol{B}_{31}$. 如果在这些步中 $\boldsymbol{B}_{32}, \boldsymbol{B}_{12}, \boldsymbol{B}_{22}$ 的数据流过 Proc(1, 2), 则可算出 $\boldsymbol{D}_{12}$:

$$\boldsymbol{D}_{12} = \boldsymbol{C}_{12} + \boldsymbol{A}_{13}\boldsymbol{B}_{32} + \boldsymbol{A}_{11}\boldsymbol{B}_{12} + \boldsymbol{A}_{12}\boldsymbol{B}_{22}.$$

同样, 在 $t = 1 : 3$ 时, 如果 Proc(1, 3) 中有 $\boldsymbol{B}_{13}, \boldsymbol{B}_{23}, \boldsymbol{B}_{33}$ 的话, 则可算出 $\boldsymbol{D}_{13} = \boldsymbol{C}_{13} + \boldsymbol{A}_{11}\boldsymbol{B}_{13} + \boldsymbol{A}_{12}\boldsymbol{B}_{23} + \boldsymbol{A}_{13}\boldsymbol{B}_{33}$. 综上所述, 设环面存储结构的初值如下:

| $\boldsymbol{A}_{11}$ | $\boldsymbol{B}_{11}$ | $\boldsymbol{A}_{12}$ | $\boldsymbol{B}_{22}$ | $\boldsymbol{A}_{13}$ | $\boldsymbol{B}_{33}$ |
|---|---|---|---|---|---|
| · | $\boldsymbol{B}_{21}$ | · | $\boldsymbol{B}_{32}$ | · | $\boldsymbol{B}_{13}$ |
| · | $\boldsymbol{B}_{31}$ | · | $\boldsymbol{B}_{12}$ | · | $\boldsymbol{B}_{23}$ |

将 $\boldsymbol{B}_{ij}$ 做向北流动, 得到

$t = 1$

| $\boldsymbol{A}_{12}$ | $\boldsymbol{B}_{21}$ | $\boldsymbol{A}_{13}$ | $\boldsymbol{B}_{32}$ | $\boldsymbol{A}_{11}$ | $\boldsymbol{B}_{13}$ |
|---|---|---|---|---|---|
| · | $\boldsymbol{B}_{31}$ | · | $\boldsymbol{B}_{12}$ | · | $\boldsymbol{B}_{23}$ |
| · | $\boldsymbol{B}_{11}$ | · | $\boldsymbol{B}_{22}$ | · | $\boldsymbol{B}_{33}$ |

$t = 2$

| $\boldsymbol{A}_{13}$ | $\boldsymbol{B}_{31}$ | $\boldsymbol{A}_{11}$ | $\boldsymbol{B}_{12}$ | $\boldsymbol{A}_{12}$ | $\boldsymbol{B}_{23}$ |
|---|---|---|---|---|---|
| · | $\boldsymbol{B}_{11}$ | · | $\boldsymbol{B}_{22}$ | · | $\boldsymbol{B}_{33}$ |
| · | $\boldsymbol{B}_{21}$ | · | $\boldsymbol{B}_{32}$ | · | $\boldsymbol{B}_{13}$ |

$t = 3$

| $\boldsymbol{A}_{11}$ | $\boldsymbol{B}_{11}$ | $\boldsymbol{A}_{12}$ | $\boldsymbol{B}_{22}$ | $\boldsymbol{A}_{13}$ | $\boldsymbol{B}_{33}$ |
|---|---|---|---|---|---|
| · | $\boldsymbol{B}_{21}$ | · | $\boldsymbol{B}_{32}$ | · | $\boldsymbol{B}_{13}$ |
| · | $\boldsymbol{B}_{31}$ | · | $\boldsymbol{B}_{12}$ | · | $\boldsymbol{B}_{23}$ |

因此, 如果把 $\boldsymbol{B}$ 按交错的顺序预先分配到环面存储结构中, 就可用第一行的处理器来计算 $\boldsymbol{C}$ 的第一行块.

如果用类似的方法将 $\boldsymbol{A}$ 的第二行和第三行也作交错放置, 则每一步可使九个处理器都进行乘–加运算. 确切地说, 如果让

| $A_{11}$ | $B_{11}$ | $A_{12}$ | $B_{22}$ | $A_{13}$ | $B_{33}$ |
|---|---|---|---|---|---|
| $A_{22}$ | $B_{21}$ | $A_{23}$ | $B_{32}$ | $A_{21}$ | $B_{13}$ |
| $A_{33}$ | $B_{31}$ | $A_{31}$ | $B_{12}$ | $A_{32}$ | $B_{23}$ |

则将 $A_{ij}$ 向西流动且 $B_{ij}$ 向北流动就得到

| $A_{12}$ | $B_{21}$ | $A_{13}$ | $B_{32}$ | $A_{11}$ | $B_{13}$ |
|---|---|---|---|---|---|
| $A_{23}$ | $B_{31}$ | $A_{21}$ | $B_{12}$ | $A_{22}$ | $B_{23}$ |
| $A_{31}$ | $B_{11}$ | $A_{32}$ | $B_{22}$ | $A_{33}$ | $B_{33}$ |

$t=1$

| $A_{13}$ | $B_{31}$ | $A_{11}$ | $B_{12}$ | $A_{12}$ | $B_{23}$ |
|---|---|---|---|---|---|
| $A_{21}$ | $B_{11}$ | $A_{22}$ | $B_{22}$ | $A_{23}$ | $B_{33}$ |
| $A_{32}$ | $B_{21}$ | $A_{33}$ | $B_{32}$ | $A_{31}$ | $B_{13}$ |

$t=2$

| $A_{11}$ | $B_{11}$ | $A_{12}$ | $B_{22}$ | $A_{13}$ | $B_{33}$ |
|---|---|---|---|---|---|
| $A_{22}$ | $B_{21}$ | $A_{23}$ | $B_{32}$ | $A_{21}$ | $B_{13}$ |
| $A_{33}$ | $B_{31}$ | $A_{31}$ | $B_{12}$ | $A_{32}$ | $B_{23}$ |

$t=3$

从这个例子我们引出一般的算法. 假设初始时 $A_{ij}, B_{ij}$ 和 $C_{ij}$ 存储于 Proc$(i,j)$ 中. 为了获得数据 $A$ 的交错安置, 注意到第 $i$ 行处理器中的 $A_{ij}$ 需向西移动 $i-1$ 个位置. 同样, 第 $j$ 列处理器中的 $B_{ij}$ 需向北移动 $j-1$ 个位置. 算法如下:

**算法 6.2.2** 假设 $A\in\mathbb{R}^{n\times n}, B\in\mathbb{R}^{n\times n}, C\in\mathbb{R}^{n\times n}$ 给定, 且满足 $D=C+AB$. 如果 $p_1\times p_1$ 环面机器的每个处理器执行如下的节点程序且有 $n=rp_1$, 则结束后 Proc$(\mu\lambda)$ 的本地变量 $C_{\text{loc}}$ 中储存 $D_{\mu}\lambda$. 假定本地内存有如下初始值: $p_1, (\mu,\lambda)$ (节点序号), north, east, south 和 west(四邻的序号), row=1+$(\mu-1)r$ : $\mu r$, col $=1+(\lambda-1)r:\lambda r$, $A_{\text{loc}}=A(\text{row},\text{col}), B_{\text{loc}}=B(\text{row},\text{col})$ 和 $C_{\text{loc}}=C(\text{row},\text{col})$.

```
{交错 A_{μj} 和 B_{iλ}}
for k = 1 : μ − 1
    send (A_loc, west); recv(A_loc, east)
end
for k = 1 : λ − 1
    send (B_loc, north); recv(B_loc, south)
end
for k = 1 : p_1
    C_loc = C_loc + A_loc B_loc
    send (A_loc, west)
    send (B_loc, north)
    recv (A_loc, east)
    recv (B_loc, south)
```

```
end
{A_μj 和 B_iλ 归位}
for k = 1 : μ − 1
    send (A_loc, east); recv(A_loc, west)
end
for k = 1 : λ − 1
    send (B_loc, south); recv(B_loc, north)
end
```

不难看出, 随着 $n/p_1$ 的增加, 本算法的计算–通信比趋近于零.

## 习　题

**6.2.1**　设计算法 6.2.1 的环实现.

**6.2.2**　一个上三角形矩阵可被其平方覆盖而不需额外有存储空间, 写一个动态的基于共享内存的算法.

## 本节注释与参考文献

二维数组的矩阵计算的讨论见:

H. T. Kung(1982). "Why Systolic Architectures?," *Computer* 15, 37–46.

D. P. O'Leary and G. W. Stewart(1985). "Data Flow Algorithms for Parallel Matrix Computations," *Comm. ACM* 28, 841–853.

B. Hendrickson and D. Womble(1994). "The Torus-Wrap Mapping for Dense Matrix Calculations on Massively Parallel Computers," *SIAM J. Sci, Comput.* 15, 1201–1226.

并行矩阵乘法的一些研究结果见:

L. E. Cannon(1969). *A Cellular Computer to Implement the Kalman Filter Algorithm*, Ph. D. Thesis, Montana State University.

K. H. Cheng and S. Sahni(1987). "VLSI Systems for Band Matrix Multiplication," *Parallel Computing* 4, 239–258.

G. Fox, S. W. Otto, and A. J. Hey (1987). "Matrix Algorithms on a Hypercube I: Matrix Multiplication," *Parallel Computing* 4, 17–31.

J. Berntsen(1989). "Communication Efficient Matrix Multiplication on Hypercubes," *Parallel Computing* 12, 335–342.

H. J. Jagadish and T. Kailath (1989). "A Family of New Efficient Arrays for Matrix Multiplication," *IEEE Trans. Comput.* 38, 149–155.

P. Bjørstad, F. Manne, T. Sørevik, and M. Vajteršic(1992). "Efficient Matrix Multiplication on SIMD Computers," *SIAM J. Matrix Anal. Appl.* 13, 386–401.

K. Mathur and S. L. Johnsson(1994). "Multiplication of Matrices of Arbitrary Shape on a Data Parallel Computer," *Parallel Computing* 20, 919–952.

R. Mathias(1995). "The Instability of Parallel Prefix Matrix Multiplication," *SIAM J. Sci. Comp.* 16, 955–973.

## 6.3 矩 阵 分 解

本节中我们给出两个并行 Cholesky 分解算法. 为展示基于分布式内存的分解运算是什么样的, 我们给出一个基于环结构的 gaxpy Cholesky 算法. 同时详细给出了外积形式的 Cholesky 方法之共享内存实现.

### 6.3.1 基于环的 Cholesky 分解

下面让我们看看如何将 Cholesky 分解分布到 $p$ 个处理器的环上. 出发点是关系式

$$G(\mu,\mu)G(\mu:n,\mu)=A(\mu:n,\mu)-\sum_{j=1}^{\mu-1}G(\mu,j)G(\mu:n,j)\equiv v(\mu:n).$$

这个等式是通过取 $n\times n$ 方程 $\boldsymbol{A}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 的第 $\mu$ 列得到的. 一旦得到向量 $v(\mu:n)$, 则 $G(\mu:n,\mu)$ 可通过简单数乘

$$G(\mu:n,\mu)=v(\mu:n)/\sqrt{v(\mu)}$$

来得到. 为简明起见, 假定 $n=p, A(\mu:n,\mu)$ 已存于 Proc($\mu$) 中. 执行结束时, 每个处理器用 $\boldsymbol{G}$ 的列取覆盖 $\boldsymbol{A}$ 相应的列. 对于 Proc($\mu$) 来说, 此算法需进行 $\mu-1$ 次形如

$$A(\mu:n,\mu)\leftarrow A(\mu:n,\mu)-G(\mu,j)G(\mu:n,j)$$

的 saxpy 更新, 然后是一次求平方根和一次数乘运算. 于是, Proc($\mu$) 的节点程序的大概流程如下:

```
for j = 1 : μ − 1
    从左邻接收一个 G 列
    如必要, 将接收到的 G 列的副本发送给右邻
    更新 A(μ : n, μ)
end
生成 G(μ : n, μ), 如必要, 发送给右邻
```

这样, Proc(1) 可以立即计算出 $G(1:n,1)=A(1:n.1)/\sqrt{A(1,1)}$ 并发送至 Proc(2). Proc(2) 一旦接受到此列值, 它就产生 $G(2:n,2)$ 并发送至 Proc(3). 依次类推, 按这种流水线设计, 一个处理器一旦完成其 $\boldsymbol{G}$ 列的计算, 它就可中止. 另外每个处理器是按升序 $G(1:n,1),G(2:n,2),\cdots$ 来接收 $\boldsymbol{G}$ 列的. 基于这些结论, 有

```
j = 1
while j < μ
    recv (g_loc(j : n), left)
```

```
        if μ < n
            send (g_loc(j : n), right)
        end
        A_loc(μ : n) = A_loc(μ : n) − g_loc(μ)g(μ : n)
        j = j + 1
    end
    A_loc(μ : n) = A_loc(μ : n)/sqrt(A_loc(μ))
    if μ < n
        send (A_loc(μ : n), right)
    end
```

注意, 接收的 $G$ 列的数目为 $j-1$. 当 $j=\mu$ 时, 则由 Proc($\mu$) 来产生并发送 $G(\mu:n,\mu)$.

现在将上述方案扩充至 $n$ 个处理器的情形. 有两个明显的方式来分配计算. 一种方法是让每个处理器计算一段连续存储的 $\boldsymbol{G}$ 列. 例如, 如果 $n=11, p=3, A=[a_1,\cdots,a_{11}]$, 则可将 $A$ 分配如下:

$$[\underbrace{a_1\ a_2\ a_3\ a_4}_{\text{Proc(1)}}\,|\,\underbrace{a_5\ a_6\ a_7\ a_8}_{\text{Proc(2)}}\,|\,\underbrace{a_9\ a_{10}\ a_{11}}_{\text{Proc(3)}}].$$

每个处理器然后可去计算相应的 $\boldsymbol{G}$ 列. 此方法的缺陷是, 以 Proc(1) 为例, 当 $\boldsymbol{G}$ 的第 4 列值计算出后, 该处理器就空闲, 而此时却仍有大量工作未完成.

如用轮流分配方式来分配计算, 则可获得较理想的负载平衡, 即

$$[\underbrace{a_1\ a_4\ a_7\ a_{10}}_{\text{Proc(1)}}\,|\,\underbrace{a_2\ a_5\ a_8\ a_{11}}_{\text{Proc(2)}}\,|\,\underbrace{a_3\ a_6\ a_9}_{\text{Proc(3)}}].$$

在这种方案下, Proc($\mu$) 负责计算 $G(:,\mu:p:n)$. 当一个处理器计算完它的 $\boldsymbol{G}$ 列后, 其他处理器最多只有一个 $\boldsymbol{G}$ 列未被计算. 这样当 $n/p \gg 1$ 时, 所有处理器几乎总是忙的.

现在仔细分析一下轮流分配的分布式 Cholesky 算法. 每个处理器都需要两个计数器. 计数器 $j$ 是 Proc($\mu$) 下一个要接收的 $\boldsymbol{G}$ 列的下标号. 同时处理器还应知道下一个要计算的 $\boldsymbol{G}$ 列的下标号. 注意, 如 $\mathrm{col}=\mu:p:n$, 则 Proc($\mu$) 要负责计算出 $G(:,\mathrm{col})$, 且 $L=\mathbf{length}(\mathrm{col})$ 是其需要计算的 $\boldsymbol{G}$ 列的数目. 用 $q$ 来记录求出 $\boldsymbol{G}$ 列的状态. 任何时刻, $\mathrm{col}(q)$ 是下一个要计算的 $\boldsymbol{G}$ 列的下标.

**算法 6.3.1** 假设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称正定矩阵, 且 $\boldsymbol{A}=\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ 是其 Cholesky 分解. 如果含 $p$ 个处理器的环中的每个节点执行如下的程序, 则结束时, Proc($\mu$) 将满足 $k=\mu:p:n$ 的 $G(k:n,k)$ 存储在本地数组 $A_{\mathrm{loc}}(1:n,L)$ 中, 其中 $L=\mathbf{length}(\mathrm{col}), \mathrm{col}=\mu:p:n$. 特别地, 满足 $q=1:L$ 的 $G(\mathrm{col}(q):n,\mathrm{col}(q))$ 存储在 $A_{\mathrm{loc}}(\mathrm{col}(q):n,q)$ 中. 假定本地内存有如下初始值: $p,\mu$(节点序号), left 和 right(左邻和右邻的序号), $n$ 和 $A_{\mathrm{loc}}=A(\mu:p:n)$.

```
j = 1; q = 1; col = μ : p : n; L = length(col)
while q ⩽ L
        if j = col(q)
            {形成 G(j : n, j)}
            A_loc(j : n, q) = A_loc(j : n, q)/√A_loc(j, q)
            if j < n
                send (A_loc(j : n, q), right)
            end
            j = j + 1
            {更新本地的列向量.}
            for k = q + 1 : L
                r = col(k)
                A_loc(r : n, k) = A_loc(r : n, k) − A_loc(r, q)A_loc(r : n, q)
            end
            q = q + 1
        else
            recv (g_loc(j : n), left)
            计算 α, 产生所接收 G 列的处理器之序号.
            计算 β, Proc(right) 的最终列的下标.
            if right≠ α ∧ j < β
                send (g_loc(j : n), right)
            end
            {更新本地的列向量.}
            for k = q : L
                r = col(k)
                A_loc(r : n, k) = A_loc(r : n, k) − g_loc(r)g_loc(r : n)
            end
            j = j + 1
        end
end
```

为说明指示系统的逻辑关系, 我们考虑一个 3 处理器且 $n = 10$ 的例子. 假定 3 个本地的 $q$ 值分别为 3, 2 和 2, 则相应的 $\text{col}(q)$ 值为 7, 5 和 6:

$$\underbrace{[a_1\ a_4\ \overset{\downarrow}{a_7}\ a_{10}|}_{\text{Proc(1)}}\underbrace{a_2\ \overset{\downarrow}{a_5}\ a_8\ a_{11}|}_{\text{Proc(2)}}\underbrace{a_3\ \overset{\downarrow}{a_6}\ a_9]}_{\text{Proc(3)}}$$

Proc(2) 生成第 5 个 $G$ 列并将 $q$ 值增加为 3.

需要解释何时将接受到的 $G$ 列发送到其右邻, 它需满足两个条件:

- 右邻不能是产生此 $G$ 列的处理器. 这样保证了所接收的 $G$ 列的循环可在合适时停止.
- 右邻必须还有需要计算的 $G$ 列. 否则 $G$ 列就会被送给一个已停止运行的处理器.

这些推理在分布式内存矩阵计算中是很典型的.

现在考查当 $n \gg p$ 时算法 6.3.1 的性质. 不难看出 Proc($\mu$) 执行的 flop 数为

$$F(\mu) = \sum_{k=1}^{L} 2(n - (\mu + (k-1)p))(\mu + (k-1)p) \approx \frac{n^3}{3p}.$$

每个处理器正好要对每个 $G$ 列做接收和发送操作. 利用 (6.3.1) 确定的通信开销模型, 每个处理器用于通信的开销为

$$m_\mu = \sum_{j=1}^{n} 2(\alpha_d + \beta_d(n-j)) \approx 2\alpha_d n + \beta_d n^2.$$

如果假设计算速率为每秒 $R$ 个 flop, 则算法 6.3.1 的计算–通信比约为 $(n/p)(1/3R\beta_d)$. 因此, 通信开销随 $n/p$ 的增大而显得越来越无关紧要.

### 6.3.2 共享内存的 Cholesky 分解

接下来考虑在共享内存环境下实现外积形式的 Cholesky 算法. 外积形式的 Cholesky 算法为:

```
for k = 1 : n
    A(k : n, k) = A(k : n, k)/sqrt(A(k, k))
    for j = k + 1 : n
        A(j : n, j) = A(j : n, j) - A(j : n, k)A(j, k)
    end
end
```

$j$ 循环是一个外积运算. 组成循环主体的 $n-k$ 次 saxpy 运算是独立的, 因而易于并行化. 对 $A(k:n,k)$ 的数乘可由某个处理器来完成且不会破坏负载均衡.

**算法 6.3.2** 假设 $A \in \mathbb{R}^{n\times n}$ 是对称正定矩阵, 存储于全局内存中且能被 $p$ 个处理器访问. 如果每个处理器执行如下的节点程序, 则结束时 $A$ 的下三角部分被其 Cholesky 因子覆盖. 假定本地内存有如下初始值: $n, p, \mu$ (节点序号).

```
for k = 1 : n
    if μ = 1
        v_loc(k : n) = A(k : n)
        v_loc(k : n) = v_loc(k : n)/sqrt(v_loc(k))
        A(k : n, k) = v_loc(k : n)
    end
    barrier
```

$v_{\text{loc}}(k+1:n) = A(k+1:n,k)$
**for** $j = (j+\mu):p:n$
  $\omega_{\text{loc}}(j:n) = A(j:n,j)$
  $\omega_{\text{loc}}(j:n) = \omega_{\text{loc}}(j:n) - v_{\text{loc}}(j)v_{\text{loc}}(j:n)$
  $A(j:n,j) = \omega_{\text{loc}}(j:n)$
**end**
**barrier**
**end**

$j$ 循环之前的数乘运算和外积运算相比工作量很小, 因此可分配给单个处理器来完成. 注意到算法需要两条barrier语句. 第一句保证了在 Proc(1) 产生 $G$ 的第 $k$ 列之后别的处理器才能进行第 $k$ 次外积运算. 第二句保证第 $k$ 步的所有工作都完成后才能开始第 $k+1$ 步的运算.

## 习 题

**6.3.1** 试将算法 6.3.1 写成一个分块形式. 假定 $n = rN$, 对于 $k = 1:N$, 有 (a)Proc(1) 生成 $G(:,1+(k-1)r:kr)$, (b) 让所有的处理器参与主子矩阵 $A(kr+1:n,kr+1:n)$ 的秩 $r$ 更新, 参见 4.2.6 节. 如每个处理机偏好 3 级运算, 则分块可以改善性能.

**6.3.2** 改写算法 6.3.2 为共享内存的 $QR$ 分解算法. Proc(1) 应生成 Householder 向量, 所有处理器都应能共享最近更新的 Householder 矩阵.

## 本节注释与参考文献

分布式内存机上矩阵分解算法的一般性描述见:

G. A. Geist and M. T. Heath(1986). "Matrix Factorization on a Hypercube," in M. T. Heath(ed)(1986). *Proceedings of First SIAM Conference on Hypercube Multiprocessors*, SIAM Publications, Philadelphia, Pa.

I. C. F. Ipsen, Y. Saad, and M. Schultz(1986). "Dense Linear Systems on a Ring of Processors," *Lin. Alg. and Its Applic.* 77, 205–239.

D. P. O'Leary and G. W. Stewart (1986). "Assignment and Scheduling in Parallel Matrix Factorization," *Lin. Alg. and Its Applic.* 77, 275–300.

R. S. Schreiber(1988). "Block Algorithms for Parallel Machines," in *Numerical Algorithms for Modern Parallel Computer Architectures*, M. H. Schultz(ed), IMA Volumes in Mathematics and Its Applications, Number 13, Springer-Verlag, Berlin, 197–207.

S. L. Johnsson and W. Lichtenstein(1993). "Block Cyclic Dense Linear Algebra," *SIAM J. Sci. Comp.* 14, 1257–1286.

专论 LU 分解、Cholesky 分解和 QR 分解的文章包括:

R. N. Kapur and J. C. Browne(1984). "Techniques for Solving Block Tridiagonal Systems on Reconfigurable Array Computers," *SIAM J. Sci. and Stat. Comp.* 5, 701–719.

G. J. Davis(1986). "Column LU Pivoting on a Hypercube Multiprocessor," *SIAM J. Alg. and Disc. Methods* 7, 538–550.

J. M. Delosme and I. C. F. Ipsen(1986). "Parallel Solution of Symmetric Positive Definite Systems with Hyperbolic Rotations," *Lin. Alg. and Its Applic.* 77, 75–112.

A. Pothen, S. Jha, and U. Vemapulati(1987). "Orthogonal Factorization on a Distributed Memory Multiprocessor," in *Hypercube Multiprocessors*, ed. M. T. Heath, SIAM Press, 1987.

C. H. Bischof(1988). "QR Factorization Algorithms for Coarse Grain Distributed Systems," PhD Thesis, Dept. of Computer Science, Cornell University, Ithaca, NY.

G. A. Geist and C. H. Romine(1988). "LU Factorization Algorithms on Distributed Memory Multiprocessor Architectures," *SIAM J. Sci, and Stat. Comp.* 9, 639–649.

J. M. Ortega and C. H. Romine(1988). "The *ijk* Forms of Factorization Methods II: Parallel Systems," *Parallel Computing* 7, 149–162.

M. Marrakchi and Y. Robert(1989). "Optimal Algorithms for Gaussian Elimination on an MIMD Computer," *Parallel Computing* 12, 183–194.

求解三角方程组的并行算法见:

R. Montoye and D. Laurie(1982). "A Practical Algorithm for the Solution of Triangular Systems on a Parallel Processing System," *IEEE Trans. Comp.* *C*-31, 1076–1082.

D. J. Evans and R. Dunbar(1983). "The Parallel Solution of Triangular Systems of Equations," *IEEE Trans. Comp.* *C*-32, 201–204.

C. H. Romine and J. M. Ortega(1988). "Partallel Solution of Triangular Systems of Equations," *Parallel Computing* 6, 109–114.

M. T. Heath and C. H. Romine(1988). "Paratllel Solution of Triangular Systems on Distributed Memory Multiprocessors," *SIAM J. Sci. and Stat. Comp.* 9, 558–588.

G. Li and T. Coleman(1988). "A Parallel Triangular Solver for a Distributed-Memory Multiprocessor," *SIAM J. Sci. and Stat. Comp.* 9, 485–502.

S. C. Eisenstat. M. T. Heath, C. S. Henkel, and C. H. Romine(1988). "Modified Cyclic Algorithms for Solving Triangular Systems on Distributed Memory Mutliprocessors," *SIAM J. Sci. and Stat. Comp.* 9, 589–600.

N. J. Higham(1995). "Stability of Parallel Triangular System Solvers," *SIAM J. Sci. Comp.* 16, 400–413.

LU 分解和 Cholesky 分解的并行计算方面的文章包括:

R. P. Brent and F. T. Luk(1982)."Computing the Cholesky Factorization Using a Systolic Architecture," *Proc. 6th Australian Computer Science Conf.* 295–302.

D. P. O'Leary and G. W. Stewart(1985). "Data Flow Algorithms for Parallel Matrix Computations," *Comm. of the ACM* 28, 841–853.

J. M. Delosme and I. C. F. Ipsen(1986). "Parallel Solution of Symmetric Positive Definite Systems with Hyperbolic Rotations," *Lin: Alg. and Its Applic.* 77, 75–112.

R. E. Funderlic and A. Geist(1986). "Torus Data Flow for Parallel Computation of Missized Matrix Problems," *Lin. Alg. and Its Applic.* 77, 149–164.

M. Costnard, M. Marrakchi, and Y. Robert(1988). "Parallel Gaussian Elimination on an MIMD Computer," *Parallel Computing* 6, 275–296.

带状矩阵和稀疏矩阵的并行算法见:

S. L. Johnsson(1985). "Solving Narrow Banded Systems on Ensemble Architectures," *ACM Trans. Math. Soft.* 11, 271–288.

S. L. Johnsson(1986). "Band Matrix System Solvers on Ensemble Architectures," in *Supercomputers: Algorithms, Architectures, and Scientific Computation*, eds. F. A. Matsen and T. Tajima, University of Texas Press, Austin TX., 196–216.

S. L. Johnsson(1987). "Solving Tridiagonal Systems on Ensemble Architectures," *SIAM J. Sci. and Stat. Comp.* 8, 354–392.

U. Meier(1985). "A Parallel Partition Method for Solving Banded Systems of Linear Equations," *Parallel Computers* 2, 33–43.

H. van der Vorst(1987). "Large Tridiagonal and Block Tridiagonal Linear Systems on Vector and Parallel Computers," *Parallel Comput.* 5, 45–54.

R. Bevilacqua, B. Codenotti, and F. Romani(1988). "Parallel Solution of Block Tridiagonal Linear Systems," *Lin. Alg. and Its Applic.* 104, 39–57.

E. G allopoulos and Y. Saad(1989). "A Parallel Block Cyclic Reduction Algorithm for the Fast Solution of Elliptic Equations," *Parallel Computing* 10, 143–160.

J. M. Conroy(1989). "A Note on the Parallel Cholesky Factorization of Wide Banded Matrices," *Parallel Computing* 10, 239–246.

M. Hegland(1991). "On the Parallel Solution of Tridiagonal Systems by Wrap-Around Partitioning and Incomplete LU Factorization," *Numer. Math.* 59, 453–472.

M. T. Heath, E. Ng, and B. W. Peyton(1991). "Parallel Algorithms for Sparse Linear Systems," *SIAM Review* 33, 420–460.

V. Mehrmann(1993). "Divide and Conquer Methods for Block Tridiagonal Systems," *Parallel Computing* 19, 257–280.

P. Raghavan(1995). "Distributed Sparse Gaussian Elimination and Orthogonal Factorization," *SIAM J. Sci. Comp.* 16, 1462–1477.

并行 QR 分解在实时信号处理中有重要应用, 细节请见:

W. M. Gentleman and H. T. Kung(1981). "Matrix Triangularization by Systolic Arrays," SPIE Proceedings, Vol. 298, 19–26.

D. E. Heller and I. C. F. Ipsen(1983). "Systolic Networks for Orthogonal Decompositions," *SIAM J. Sci. and Stat. Comp.* 4, 261–269.

M. Costnard, J. M. Muller, and Y. Robert(1986). "Parallel QR Decomposition of a Rectangular Matrix," *Numer. Math.* 48, 239–250.

L. Eldin and R. Schreiber(1986). "An Application of Systolic Arrays to Linear Discrete Ill-Posed Problems," *SIAM J. Sci. and Stat. Comp.* 7, 892–903.

F. T. Luk(1986). "A Rotation Method for Computing the QR Factorization," *SIAM J. Sci. and Stat. Comp.* 7, 452–459.

J. J. Modi and M. R. B. Clarke(1986). " An Alternative Givens Ordering," *Numer. Math.*

43, 83–90.

P. Amodio and L. Brugnano(1995). "The Parallel QR Factorization Algorithm for Tridiagonal Linear Systems," *Parallel Computing* 21, 1097–1110.

在共享内存机器上的并行分解算法在下述文章中得到详细讨论：

S. Chen, D. Kuck, and A. Sameh(1978). "Practical Parallel Band Triangular Systems Solvers," *ACM Trans. Math. Soft.* 4, 270–277.

A. Sameh and D. Kuck(1978). "On Stable Parallel Linear System Solvers," *J. Assoc. Comp. Mach.* 25, 81–91.

P. Swarztrauber(1979). "A Parallel Algorithm for Solving General Tridiagonal Equations," *Math. Comp.* 33, 185–199.

S. Chen, J. Dongarra, and C. Hsuing(1984). "Multiprocessing Linear Algebra Agorithms on the Cray X-MP-2: Experiences with Small Granularity," *J. Parallel and Distributed Computing* 1, 22–31.

J. J. Dongarra and A. H. Sameh(1984). "On Some Parallel Banded System Solvers," *Parallel Computing* 1, 223–235.

J. J. Dongarra and R. E. Hiromoto(1984). "A Collection of Parallel Linear Equation Routines for the Denelcor HEP," *Parallel Computing* 1, 133–142.

J. J. Dongarra and T. Hewitt(1986). "Implementing Dense Linear Algebra Algorithms Using Multitasking on the Cray X-MP-4(or Approaching the Gigaflop)," *SIAM J. Sci. and Stat. Comp.* 7, 347–350.

J. J. Dongarra, A. Sameh, and D. Sorensen(1986). "Implementation of Some Concurrent Algorithms for Matrix Factorization," *Parallel Computing* 3, 25–34.

A. George, M. T. Heath, and J. Liu(1986). "Parallel Cholesky Factorization on a Shared Memory Multiprocessor," *Lin. Alg. Land Its Applic.* 77, 165–187.

J. J. Dongarra and D. C. Sorensen(1987). "Linear Algebra on High Performance Computers," *Appl. Math. and Comp.* 20, 57–88.

K. Dackland, E. Elmroth, and B. Kagstrom(1992). "Parallel Block Factorizations on the Shared Memory Multiprocessor IBM 3090 VF/600J," *International J. Supercomputer Applications*, 6, 69–97.

# 第 7 章　非对称特征值问题

讨论完线性方程组和最小二乘法后, 我们把注意力转移到矩阵计算的第三个大问题: 代数特征值问题. 本章讨论非对称矩阵的特征值问题, 第 8 章讨论较易的对称矩阵情形.

我们首先介绍特征值和不变子空间的基本性质, 以及 Schur 分解和 Jordan 分解. 这两种分解的性能对比放在 7.2 节中讨论, 在那里, 我们要考察特征值和不变子空间是如何受扰动影响的, 该节还将对条件数加以研究, 以便对舍入过程中产生的误差进行估计.

本章的主要算法就是著名的 QR 算法, 它是本书中最复杂的算法, 我们用了三节的篇幅来叙述它. 作为简单的幂法的自然推广, 我们在 7.3 节中导出基本的 QR 迭代. 随后的两节着重在计算上实现这种基本迭代, 这涉及 Hessenberg 分解 (7.4 节) 及原点位移的概念 (7.5 节).

QR 算法计算矩阵的实 Schur 型, 这是一种显示特征值而非特征向量的典范型. 因而, 如果想了解不变子空间的信息, 还需要额外的计算. "计算了实 Schur 型后怎么办?" 可做为 7.6 节的副标题. 在此节我们将讨论各种不变子空间的计算, 这些计算可接着 QR 算法进行.

在 7.7 节, 我们讨论广义特征值问题 $\boldsymbol{Ax} = \lambda \boldsymbol{Bx}$ 以及为解决此问题而设计的 QR 算法的变形, 称之为 QZ 算法. 这种算法强调了正交矩阵在特征问题 (本章的中心议题) 中的重要性.

此时, 评价一下复运算和实运算是适当的. 在本书中, 我们集中精力讨论实矩阵问题的实算法, 本章也不例外, 尽管是一个实的非对称矩阵可能有复的特征值. 然而, 实用的实 QR 算法的推导及特征问题本身的数学分析, 在复数域中进行是很方便的. 因而, 读者会发现我们在 7.1 节 ~7.3 节中已换用复数记号了. 在这几节里, 我们运用 QR 分解、奇异值分解及 CS 分解的复数形式.

**预备知识**

阅读本章需要掌握第 1~3 章和 5.1 节 ~5.2 节的知识. 本章各节间的关系如下:

$$7.1\text{节} \rightarrow 7.2\text{节} \rightarrow 7.3\text{节} \rightarrow 7.4\text{节} \rightarrow 7.5\text{节} \rightarrow 7.6\text{节} \rightarrow 7.7\text{节}$$

补充参考书有: Fox(1964), Wilkinson(1965), Gourlay and Watson (1973), Stewart (1973), Hager(1988), Ciarlet(1989), Stewart and Sun(1990), Watkins(1991), Saad (1992), Jennings and Mc Keowen(1992), Datta(1995), Trefethen and Bau(1997), and Demmel(1996). 与本章有关的一些重要的 MATLAB 函数有: `eig`, `poly`, `polyeig`,

`hess`, `qz`, `rsf2csf`, `cdf2rdf`, `schur` 及`balance`. 以下是与 LAPACK 相关的例程:

| LAPACK: 非对称特征值问题 | |
|---|---|
| _GEBAL | 平衡变换 |
| _GEBAK | 取消平衡变换 |
| _GEHRD | Hessenberg 约化 $U^{\mathrm{H}}AV = H$ |
| _ORMHR | $U$(分解型) 乘矩阵 (实型) |
| _ORGHR | 产生$U$(实型) |
| _UNMHR | $U$(分解型) 乘以矩阵 (复型) |
| _UNGHR | 产生 $U$(复型) |
| _HSEQR | Hessenberg 矩阵的 Schur 分解 |
| _HSEIN | 用逆迭代法求 Hessenberg 矩阵的特征向量 |
| _GEES | 一般矩阵的 Schur 分解, 特征值按序排列 |
| _GEESX | 同上, 但给出条件数估计 |
| _GEEV | 一般矩阵的特征值和左右特征向量 |
| _GEEVX | 同上, 但给出条件数估计 |
| _TREVC | 上拟三角形矩阵的选定特征向量 |
| _TRSNA | 上拟三角形矩阵的选定特征值的条件数估计 |
| _TREXC | Schur 分解的酉重排列 |
| _TRSEM | 同上, 但给出条件数估计 |
| _TRSYL | 求解 $AX + XB = C$, $A$ 和 $B$ 为上拟三角形矩阵 |

| LAPACK: 非对称的广义特征问题 | |
|---|---|
| _GGBAL | 平衡变换 |
| _GGHRD | 约化为 Hessenberg 三角形型 |
| _HGEQZ | 广义 Schur 分解 |
| _TGEVC | 特征向量 |
| _GGBAK | 取消平衡变换 |

# 7.1 性质与分解

本节我们给出必要的数学基础知识, 以便导出和分析后面的求特征值的算法.

## 7.1.1 特征值和不变子空间

矩阵 $A \in \mathbb{C}^{n\times n}$ 的特征值是其特征多项式$p(z)$ =det$(zI - A)$ 的 $n$ 个根. 这些根的集合称为谱, 记为 $\lambda(A)$. 如果 $\lambda(A) = \{\lambda_1, \cdots, \lambda_n\}$, 则有

$$\det(A) = \lambda_1\lambda_2\cdots\lambda_n.$$

而且, 如果我们定义 $A$ 的迹为

$$\mathrm{tr}(A) = \sum_{i=1}^{n} a_{ii},$$

则 $\mathrm{tr}(A) = \lambda_1 + \cdots + \lambda_n$. 这个结论可以通过观察特征多项式中 $z^{n-1}$ 的系数得出.

如果 $\lambda \in \lambda(A)$, 那么满足

$$Ax = \lambda x$$

的非零向量 $\boldsymbol{x} \in \mathbb{C}^n$ 称为**特征向量**. 更精确地说, 满足 $\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$ 的非零向量 $\boldsymbol{x}$ 称为关于 $\lambda$ 的**右特征向量**, 满足 $\boldsymbol{x}^{\mathrm{H}}\boldsymbol{A} = \lambda \boldsymbol{x}^{\mathrm{H}}$ 的非零向量称为关于 $\lambda$ 的**左特征向量**. 除非特别声明, "特征向量" 指 "右特征向量".

一个特征向量定义了一个一维子空间, 这个子空间用矩阵 $\boldsymbol{A}$ 左乘保持不变性. 更一般地, 一个子空间 $S \subseteq \mathbb{C}^n$, 若有性质

$$\boldsymbol{x} \in S \Longrightarrow \boldsymbol{A}\boldsymbol{x} \in S,$$

则称为**不变的**(关于 $\boldsymbol{A}$). 注意到, 如果

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}\boldsymbol{B},\ \boldsymbol{B} \in \mathbb{C}^{k\times k},\ \boldsymbol{X} \in \mathbb{C}^{n\times k},$$

那么 $\mathrm{ran}(\boldsymbol{X})$ 是不变的, 且 $\boldsymbol{B}\boldsymbol{y} = \lambda y \Longrightarrow \boldsymbol{A}(\boldsymbol{X}\boldsymbol{y}) = \lambda(\boldsymbol{X}\boldsymbol{y})$. 这样, 如果 $\boldsymbol{X}$ 是列满秩的, 那么 $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}\boldsymbol{B}$ 隐含有 $\lambda(\boldsymbol{B}) \subseteq \lambda(\boldsymbol{A})$. 如果 $\boldsymbol{X}$ 是方阵且非奇异, 那么 $\lambda(\boldsymbol{A}) = \lambda(\boldsymbol{B})$ 且我们称 $\boldsymbol{A}$ 和 $\boldsymbol{B} = \boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}$**相似**. 在此意义下, 称 $\boldsymbol{X}$ 为一个**相似变换**.

### 7.1.2 解耦

许多特征值的计算需要把一个给定的问题分解为若干小的特征问题来逐一解决. 下面结论是这些约化的基础.

**引理 7.1.1** *如果 $\boldsymbol{T} \in \mathbb{C}^{n\times n}$ 的划分如下*

$$\boldsymbol{T} = \begin{array}{c} \left[\begin{array}{cc} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{array}\right] \begin{array}{c} p \\ q \end{array} \\ \begin{array}{cc} p & \quad q \end{array} \end{array}$$

*那么 $\lambda(\boldsymbol{T}) = \lambda(\boldsymbol{T}_{11}) \cup \lambda(\boldsymbol{T}_{22})$.*

**证明** 设

$$\boldsymbol{T}\boldsymbol{x} = \left[\begin{array}{cc} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{array}\right] \left[\begin{array}{c} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{array}\right] = \lambda \left[\begin{array}{c} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{array}\right],$$

这里 $\boldsymbol{x}_1 \in \mathbb{C}^p$ 且 $\boldsymbol{x}_2 \in \mathbb{C}^q$. 如果 $\boldsymbol{x}_2 \neq 0$, 那么 $\boldsymbol{T}_{22}\boldsymbol{x}_2 = \lambda \boldsymbol{x}_2$, 因此 $\lambda \in \lambda(\boldsymbol{T}_{22})$. 如果 $\boldsymbol{x}_2 = 0$, 那么 $\boldsymbol{T}_{11}\boldsymbol{x}_1 = \lambda \boldsymbol{x}_1$, 这样 $\lambda \in \lambda(\boldsymbol{T}_{11})$. 由此可知 $\lambda(\boldsymbol{T}) \subset \lambda(\boldsymbol{T}_{11}) \cup \lambda(\boldsymbol{T}_{22})$. 但由于集合 $\lambda(\boldsymbol{T})$ 和 $\lambda(\boldsymbol{T}_{11}) \cup \lambda(\boldsymbol{T}_{22})$ 有相同的基数, 所以它们是相等的. □

### 7.1.3 基本酉相似分解

利用相似变换, 可以将一个给定矩阵约化为几种典范型的任一个. 这些典范型因显示特征值的方式以及提供的不变子空间信息形式而异. 考虑到数值稳定性, 我们先讨论能用酉相似变换做的约化.

**引理 7.1.2** *如果 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$, $\boldsymbol{B} \in \mathbb{C}^{p\times p}$, 且 $\boldsymbol{X} \in \mathbb{C}^{n\times p}$ 满足*

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}\boldsymbol{B},\ \mathrm{rank}(\boldsymbol{X}) = p, \tag{7.1.1}$$

*那么存在一个酉矩阵 $\boldsymbol{Q} \in \mathbb{C}^{n\times n}$, 使得*

$$Q^{\mathrm{H}}AQ = T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{matrix} p \\ n-p \end{matrix} \tag{7.1.2}$$
$$\qquad\qquad\quad p \qquad n-p$$

这里 $\lambda(T_{11}) = \lambda(A) \cap \lambda(B)$.

**证明** 设

$$X = Q \begin{bmatrix} R_1 \\ 0 \end{bmatrix}, \quad Q \in \mathbb{C}^{n\times n},\ R_1 \in \mathbb{C}^{p\times p}$$

是 $X$ 的一个 QR 分解. 将之代入 (7.1.1) 并整理得

$$\begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} B,$$

这里

$$Q^{\mathrm{H}}AQ = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{matrix} p \\ n-p \end{matrix}$$
$$\qquad\qquad p \qquad n-p$$

利用 $R_1$ 的非奇异性以及方程 $T_{21}R_1 = 0$ 和 $T_{11}R_1 = R_1B$, 我们可以推出 $T_{21} = 0$ 和 $\lambda(T_{11}) = \lambda(B)$. 由引理 7.1.1, $\lambda(A) = \lambda(T) = \lambda(T_{11}) \cup \lambda(T_{22})$, 结论得证. □

**例 7.1.1** 如果

$$A = \begin{bmatrix} 67.00 & 177.60 & -63.20 \\ -20.40 & 95.88 & -87.16 \\ 22.80 & 67.84 & 12.12 \end{bmatrix},$$

$X = (20, -9 - 12)^{\mathrm{T}}$ 且 $B = [25]$, 那么 $AX = XB$. 而且, 如果正交矩阵 $Q$ 定义为

$$Q = \begin{bmatrix} -0.800 & 0.360 & 0.480 \\ 0.360 & 0.928 & -0.096 \\ 0.480 & -0.096 & 0.872 \end{bmatrix},$$

那么 $Q^{\mathrm{T}}X = (-25, 0, 0)^{\mathrm{T}}$ 且

$$Q^{\mathrm{T}}AQ = T = \begin{bmatrix} 25 & -90 & 5 \\ 0 & 147 & -104 \\ 0 & 146 & 3 \end{bmatrix}$$

由此易知 $\lambda(A) = \{15, 75 + 100\mathrm{i}, 75 - 100\mathrm{i}\}$.

引理 7.1.2 是说, 如果我们知道某个矩阵的不变子空间, 则可应用酉相似变换将该矩阵约化为分块三角形型. 利用归纳法可容易地建立 Schur 分解定理 (1909).

**定理 7.1.3(Schur 分解)** 如果 $A \in \mathbb{C}^{n\times n}$, 那么存在一个酉矩阵 $Q \in \mathbb{C}^{n\times n}$, 使得

$$Q^{\mathrm{H}}AQ = T = D + N, \tag{7.1.3}$$

这里 $D = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$, $N \in \mathbb{C}^{n\times n}$ 是严格上三角形矩阵. 进一步可以选取 $Q$ 使得特征值 $\lambda_i$ 沿对角线按任一给定的次序出现.

**证明** 当 $n = 1$ 时, 定理显然成立. 假设定理对于所有的阶数为 $n-1$ 或少于 $n-1$ 的矩阵均成立. 如果 $Ax = \lambda x$, 这里 $x \neq 0$, 那么由引理 7.12(这里 $B = (\lambda)$) 存在一个酉矩阵 $U$ 使得

$$U^{\mathrm{H}}AU = \begin{bmatrix} \lambda & w^{\mathrm{H}} \\ 0 & C \end{bmatrix} \begin{matrix} 1 \\ n-1 \end{matrix}. \\ \quad\ \ \begin{matrix} 1 & n-1 \end{matrix}$$

由归纳法假设存在一个酉矩阵 $\tilde{U}$ 使得 $\tilde{U}^{\mathrm{H}}C\tilde{U}$ 是上三角形矩阵. 这样, 如果 $Q = U\,\mathrm{diag}(1, \tilde{U})$, 那么 $Q^{\mathrm{H}}AQ$ 是上三角形矩阵. □

**例 7.1.2** 如果

$$A = \begin{bmatrix} 3 & 8 \\ -2 & 3 \end{bmatrix}, \quad Q = \begin{bmatrix} 0.8944\mathrm{i} & 0.4472 \\ -0.4472 & -0.8944\mathrm{i} \end{bmatrix},$$

那么 $Q$ 是酉矩阵且

$$Q^{\mathrm{H}}AQ = \begin{bmatrix} 3+4\mathrm{i} & -6 \\ 0 & 3-4\mathrm{i} \end{bmatrix}$$

如果 $Q = [q_1, \cdots, q_n]$ 是 (7.1.3) 中的酉矩阵 $Q$ 的列分块形式, 则 $q_i$ 称为 Schur 向量. 令等式 $AQ = QT$ 两边的列向量相等, 我们知道 Schur 向量满足

$$Aq_k = \lambda_k q_k + \sum_{i=1}^{k-1} n_{ik} q_i, \quad k = 1:n. \tag{7.1.4}$$

由此我们得知子空间

$$S_k = \mathrm{span}\{q_1, \cdots, q_k\}, \quad k = 1:n$$

是不变的. 而且, 不难证明, 如果 $Q_k = [q_1, \cdots, q_k]$, 那么 $\lambda(Q_k^{\mathrm{H}}AQ_k) = \{\lambda_1, \cdots, \lambda_k\}$. 由于 (7.1.3) 中的特征值可以随意排序, 故对应于每 $k$ 个特征值所组成的子集都至少存在一个 $k$ 维不变子空间.

从 (7.1.4) 还可得出另外一个结论, 即 Schur 向量 $q_k$ 是一个特征向量当且仅当 $N$ 的第 $k$ 列为零向量. 对于 $k = 1:n$, 当 $A^{\mathrm{H}}A = AA^{\mathrm{H}}$ 时, 这种情形都出现. 此时称 $A$ 为正规矩阵.

**推论 7.1.4** $A \in \mathbb{C}^{n\times n}$ 是正规矩阵当且仅当存在一个酉矩阵 $Q \in \mathbb{C}^{n\times n}$ 使得

$$Q^{\mathrm{H}}AQ = \mathrm{diag}(\lambda_1, \cdots, \lambda_n).$$

**证明** 易知, 如果 $\boldsymbol{A}$ 酉相似于一个对角矩阵, 那么 $\boldsymbol{A}$ 是正规矩阵. 另一方面, 如果 $\boldsymbol{A}$ 正规且 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 是它的 Schur 分解, 那么 $\boldsymbol{T}$ 也正规. 而一个正规的上三角形矩阵 $\boldsymbol{T}$ 必为对角矩阵. 推论证毕. □

注意到, 如果 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}=\mathrm{diag}(\lambda_i)+\boldsymbol{N}$ 是一般 $n\times n$ 矩阵 $\boldsymbol{A}$ 的 Schur 分解, 那么 $\|\boldsymbol{N}\|_F$ 与 $\boldsymbol{Q}$ 的选择无关:

$$\|\boldsymbol{N}\|_F^2=\|\boldsymbol{A}\|_F^2-\sum_{i=1}^{n}|\lambda_i|^2\equiv\Delta^2(\boldsymbol{A}).$$

这个量称为 $\boldsymbol{A}$ 的*正规偏离度*. 这样, 要使 $\boldsymbol{T}$"更对角化", 就必须利用非酉相似变换.

### 7.1.4 非酉约化

为了说明什么是非相似约化, 我们考虑一个 $2\times2$ 的块三角形矩阵的分块对角化.

**引理 7.1.5** *设 $\boldsymbol{T}\in\mathbb{C}^{n\times n}$ 的划分如下:*

$$\boldsymbol{T}=\begin{bmatrix}\boldsymbol{T}_{11} & \boldsymbol{T}_{12}\\ \boldsymbol{0} & \boldsymbol{T}_{12}\end{bmatrix}\begin{matrix}p\\ q\end{matrix}.\qquad \begin{matrix}p & q\end{matrix}$$

*定义线性变换 $\phi:\mathbb{C}^{p\times q}\to\mathbb{C}^{p\times q}$ 为*

$$\phi(\boldsymbol{X})=\boldsymbol{T}_{11}\boldsymbol{X}-\boldsymbol{X}\boldsymbol{T}_{22},$$

*这里 $\boldsymbol{X}\in\mathbb{C}^{p\times q}$. 那么 $\phi$ 是非奇异的当且仅当 $\lambda(\boldsymbol{T}_{11})\cap\lambda(\boldsymbol{T}_{22})=\varnothing$. 若 $\phi$ 非奇异且 $\boldsymbol{Y}$ 定义为*

$$\boldsymbol{Y}=\begin{bmatrix}\boldsymbol{I}_p & \boldsymbol{Z}\\ \boldsymbol{0} & \boldsymbol{I}_q\end{bmatrix},\quad \phi(\boldsymbol{Z})=-\boldsymbol{T}_{12},$$

*那么 $\boldsymbol{Y}^{-1}\boldsymbol{T}\boldsymbol{Y}=\mathrm{diag}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})$.*

**证明** 假设对 $\boldsymbol{X}\neq\boldsymbol{0}$ 有 $\phi(\boldsymbol{X})=\boldsymbol{0}$ 且

$$\boldsymbol{U}^{\mathrm{H}}\boldsymbol{X}\boldsymbol{V}=\begin{bmatrix}\boldsymbol{\Sigma}_r & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{matrix}r\\ p-r\end{matrix}\qquad \begin{matrix}r & q-r\end{matrix}$$

是 $\boldsymbol{X}$ 的 SVD 分解, 其中 $\boldsymbol{\Sigma}_r$=diga$(\sigma_i)$, $r=\mathrm{rank}(\boldsymbol{X})$. 将之代入方程 $\boldsymbol{T}_{11}\boldsymbol{X}=\boldsymbol{X}\boldsymbol{T}_{22}$ 得到

$$\begin{bmatrix}\boldsymbol{A}_{11} & \boldsymbol{A}_{12}\\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Sigma}_r & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}=\begin{bmatrix}\boldsymbol{\Sigma}_r & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{B}_{11} & \boldsymbol{B}_{12}\\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22}\end{bmatrix},$$

其中 $\boldsymbol{U}^{\mathrm{H}}\boldsymbol{T}_{11}\boldsymbol{U}=(\boldsymbol{A}_{ij})$ 和 $\boldsymbol{V}^{\mathrm{H}}\boldsymbol{T}_{22}\boldsymbol{V}=(\boldsymbol{B}_{ij})$. 通过分块比较我们知道 $\boldsymbol{A}_{21}=\boldsymbol{0}$, $\boldsymbol{B}_{12}=\boldsymbol{0}$ 且 $\lambda(\boldsymbol{A}_{11})=\lambda(\boldsymbol{B}_{11})$. 故

$$\varnothing\neq\lambda(\boldsymbol{A}_{11})=\lambda(\boldsymbol{B}_{11})\subseteq\lambda(\boldsymbol{T}_{11})\cap\lambda(\boldsymbol{T}_{22}).$$

另一方面, 如果 $\lambda \in \lambda(\boldsymbol{T}_{11}) \cap \lambda(\boldsymbol{T}_{22})$, 那么我们有非零向量 $x$ 和 $y$, 使得 $\boldsymbol{T}_{11}x = \lambda x$ 且 $y^{\mathrm{H}}\boldsymbol{T}_{22} = \lambda y^{\mathrm{H}}$. 由计算知 $\phi(xy^{\mathrm{H}}) = 0$. 最后, 如果 $\phi$ 非奇异, 那么以上定义的矩阵 $\boldsymbol{Z}$ 存在且

$$\boldsymbol{Y}^{-1}\boldsymbol{T}\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{Z} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{Z} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{11}\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{T}_{22} + \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{bmatrix}. \quad \square$$

**例 7.1.3** 若

$$\boldsymbol{T} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 3 & 8 \\ 0 & -2 & 3 \end{bmatrix} \text{且 } \boldsymbol{Y} = \begin{bmatrix} 1.0 & 0.5 & -0.5 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix},$$

则

$$\boldsymbol{Y}^{-1}\boldsymbol{T}\boldsymbol{Y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 8 \\ 0 & -2 & 3 \end{bmatrix}.$$

通过重复应用引理 7.1.5, 我们能建立以下更一般的结论.

**定理 7.1.6(分块对角分解)** 假设

$$\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} & \cdots & \boldsymbol{T}_{1q} \\ 0 & \boldsymbol{T}_{22} & \cdots & \boldsymbol{T}_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{T}_{qq} \end{bmatrix} \tag{7.1.5}$$

是 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ 的一个 Schur 分解且设 $T_{ii}$ 为方阵. 如果当 $i \neq j$ 时 $\lambda(\boldsymbol{T}_{ii}) \bigcap \lambda(\boldsymbol{T}_{jj}) = \varnothing$, 那么存在一个非奇异矩阵 $\boldsymbol{Y} \in \mathbb{C}^{n\times n}$ 使得

$$(\boldsymbol{Q}\boldsymbol{Y})^{-1}\boldsymbol{A}(\boldsymbol{Q}\boldsymbol{Y}) = \mathrm{diag}(\boldsymbol{T}_{11}, \cdots, \boldsymbol{T}_{qq}). \tag{7.1.6}$$

**证明** 利用引理 7.1.5 及归纳法即可获证. $\square$

若每个对角块 $\boldsymbol{T}_{ii}$ 对应不同的特征值, 那么我们有以下推论.

**推论 7.1.7** 如果 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$, 那么存在一非奇异矩阵 $\boldsymbol{X}$ 使得

$$\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} = \mathrm{diag}(\lambda_1\boldsymbol{I} + \boldsymbol{N}_1, \cdots, \lambda_q\boldsymbol{I} + \boldsymbol{N}_q),\ \boldsymbol{N}_i \in \mathbb{C}^{n_i\times n_i}, \tag{7.1.7}$$

这里 $\lambda_1, \cdots \lambda_q$ 互不相等, 整数 $n_1 \cdots, n_q$ 满足 $n_1 + \cdots + n_q = n$, 且每个 $\boldsymbol{N}_i$ 都是严格上三角形矩阵.

许多重要的术语与分解式 (7.1.7) 有关. 整数 $n_i$ 称为 $\lambda_i$ 的代数重数. 如果 $n_i = 1$, 那么称 $\lambda_i$ 为单特征值. $\lambda_i$ 的几何重数等于 $\mathrm{null}(\boldsymbol{N}_i)$ 的维数, 即与 $\lambda_i$ 相对

应的线性无关的特征向量之个数. 如果 $\lambda_i$ 的代数重数大于它的几何重数, 那么称 $\lambda_i$ 为退化的特征值. 如果一个矩阵有退化特征值, 则称它为退化矩阵. 基于下面的结论, 非退化矩阵也称为可对角化矩阵.

**推论 7.1.8(对角型)** $A \in \mathbb{C}^{n\times n}$ 非退化当且仅当存在一个非奇异矩阵 $X \in \mathbb{C}^{n\times n}$ 使得

$$X^{-1}AX = \mathrm{diag}(\lambda_1, \cdots, \lambda_n). \tag{7.1.8}$$

**证明** $A$ 非退化当且仅当存在线性无关向量 $x_1, \cdots, x_n \in \mathbb{C}^n$ 以及标量 $\lambda_1, \cdots, \lambda_n$ 使得 $Ax_i = \lambda_i x_i$, $i = l : n$. 这等价于存在一个非奇异矩阵 $X = [x_1, \cdots, x_n] \in \mathbb{C}^{n\times n}$ 使得 $AX = XD$, 其中 $D$ =diag$(\lambda_1, \cdots, \lambda_n)$. □

注意到, 如果 $y_i^{\mathrm{H}}$ 是 $X^{-1}$ 的第 $i$ 行, 那么 $y_i^{\mathrm{H}}A = \lambda_i y_i^{\mathrm{H}}$. 这样,$X^{-\mathrm{T}}$ 的列为左特征向量, $X$ 的列为右特征向量.

**例 7.1.4** 如果

$$A = \begin{bmatrix} 5 & -1 \\ -2 & 6 \end{bmatrix}, \quad X = \begin{bmatrix} 1 & 1 \\ 1 & -2 \end{bmatrix},$$

那么 $X^{-1}AX$ =diag(4,7).

如果我们将 (7.1.7) 中的 $X$ 划分如下,

$$X = [\underset{n_1}{X_1}, \cdots, \underset{n_q}{X_q}]$$

则 $\mathbb{C}^n$ =ran$(X_1) \oplus \cdots \oplus$ran$(X_q)$, 是不变子空间的直和. 如果这些子空间的基特别选取, 那么有可能在 $X^{-1}AX$ 的上三角部分产生更多的零元素.

**定理 7.1.9(Jordan 分解)** 如果 $A \in \mathbb{C}^{n\times n}$, 那么存在一个非奇异矩阵 $X \in \mathbb{C}^{n\times n}$, 使得 $X^{-1}AX = \mathrm{diag}(J_1, \cdots, J_t)$, 这里

$$J_i = \begin{bmatrix} \lambda_i & 1 & & \cdots & 0 \\ 0 & \lambda_i & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & & 0 & \lambda_i \end{bmatrix}$$

是 $m_i \times m_i$ 矩阵且 $m_1 + \cdots + m_t = n$.

**证明** 参看 Halmos(1958, 从 112 页起.) □

这里 $J_i$ 称为 Jordan 块. 尽管 Jordan 块沿对角线的顺序不唯一, 但与每一不同特征值相对应的 Jordan 块的个数和维数是唯一确定的.

### 7.1.5 对非酉相似变换的几点注释

一个退化矩阵的 Jordan 块结构难以从数值上确定. 可对角化的 $n\times n$ 矩阵的集合在 $\mathbb{C}^{n\times n}$ 中是稠密的. 因而, 一个退化矩阵的微小扰动就会使其 Jordan 型面目全非. 我们在 7.6.5 节中对此做进一步探讨.

在特征值问题中, 一个相关困难是几乎退化的矩阵的特征向量矩阵可能是坏条件的. 例如, 任何将

$$\boldsymbol{A}=\begin{bmatrix}1+\varepsilon & 1\\ 0 & 1-\varepsilon\end{bmatrix},\quad 0<\varepsilon\ll 1 \tag{7.1.9}$$

对角化的矩阵 $\boldsymbol{X}$ 都有 $\dfrac{1}{\varepsilon}$ 量级的条件数 (2 范数意义下).

这些观察揭示了病态相似变换所导致的困难. 由于

$$fl(\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X})=\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}+\boldsymbol{E}, \tag{7.1.10}$$

其中

$$\|\boldsymbol{E}\|_2\approx\mu\kappa_2(\boldsymbol{X})\|\boldsymbol{A}\|_2. \tag{7.1.11}$$

显然, 不用酉相似变换计算特征值, 会产生很大的误差.

### 7.1.6 奇异值和特征值

由于 $\boldsymbol{A}$ 和它的 Schur 分解 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\mathrm{diag}(\lambda_i)+\boldsymbol{N}$ 有相同的奇异值, 可以得出

$$\sigma_{\min}(\boldsymbol{A})\leqslant\min_i|\lambda_i|\leqslant\max_i|\lambda_i|\leqslant\sigma_{\max}(\boldsymbol{A}).$$

从已知的关于三角形矩阵的条件数知识可知, 有可能

$$\max_{i,j}\frac{|\lambda_i|}{|\lambda_j|}\ll\kappa_2(\boldsymbol{A}).$$

这提醒我们, 对于非正规矩阵, 当分析 $\boldsymbol{A}x=b$ 的灵敏性时, 特征值没有奇异值那样的 “预示性”. 非正规矩阵的特征值有别的缺陷. 参看 11.3.4 节.

## 习　题

**7.1.1**　证明: 如果 $\boldsymbol{T}\in\mathbb{C}^{n\times n}$ 是上三角形矩阵且正规, 则 $\boldsymbol{T}$ 是对角矩阵.

**7.1.2**　证明: 如果 $\boldsymbol{X}^0$ 对角化 (7.1.9) 式中 $2\times 2$ 矩阵且 $\varepsilon\leqslant\dfrac{1}{2}$ 那么 $\kappa_1(X)\geqslant\dfrac{1}{\varepsilon}$.

**7.1.3**　假设 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 有不同的特征值, 试证明, 如果 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 是它的 Schur 分解且 $\boldsymbol{A}\boldsymbol{B}=\boldsymbol{B}\boldsymbol{A}$, 那么 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{B}\boldsymbol{Q}$ 是上三角形矩阵.

**7.1.4**　证明: 如果 $\boldsymbol{A}$ 和 $\boldsymbol{B}^{\mathrm{H}}$ 是在 $\mathbb{C}^{m\times n}$ 中且 $m\geqslant n$, 那么: $\lambda(\boldsymbol{A}\boldsymbol{B})=\lambda(\boldsymbol{B}\boldsymbol{A})\bigcup\{\underbrace{0,\cdots,0}_{m-n}\}$.

**7.1.5**　给定 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$, 用 Schur 分解证明, 对任意 $\varepsilon>0$, 存在一个可对角化的矩阵 $\boldsymbol{B}$, 使得 $\|\boldsymbol{A}-\boldsymbol{B}\|_2\leqslant\varepsilon$. 这表明可对角化矩阵的集合在 $\mathbb{C}^{n\times n}$ 中是稠密的, 并且 Jordan 标准型不是一个连续的矩阵分解.

**7.1.6**　假定 $\boldsymbol{A}_k\to\boldsymbol{A}$ 且 $\boldsymbol{Q}_k^{\mathrm{H}}\boldsymbol{A}_k\boldsymbol{Q}_k=\boldsymbol{T}_k$ 是 $\boldsymbol{A}_k$ 的 Schur 分解. 证明 $\{\boldsymbol{Q}_k\}$ 有收敛子列 $\{\boldsymbol{Q}_{k_i}\}$, 满足

$$\lim_{i\to\infty}\boldsymbol{Q}_{k_i}=\boldsymbol{Q},$$

其中 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 是上三角形矩阵. 这表明矩阵的特征值是它的元素的连续函数.

**7.1.7**　证明 (7.1.10) 和 (7.1.11).

**7.1.8** 试说明怎样计算

$$M = \begin{bmatrix} A & C \\ B & D \end{bmatrix} \begin{matrix} k \\ j \end{matrix}$$
$$\quad k \quad j$$

的特征值, 这里 $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ 和 $\boldsymbol{D}$ 是给定的实对称矩阵.

**7.1.9** 用 Jordan 标准形证明: 若矩阵 $\boldsymbol{A}$ 的所有特征值均严格小于 1, 则 $\lim\limits_{k\to\infty} \boldsymbol{A}^k = \boldsymbol{0}$.

**7.1.10** 初值问题:

$$\dot{x}(t) = y(t), \quad x(0) = 1,$$
$$\dot{y}(t) = -x(t), \quad y(0) = 0$$

有解 $x(t) = \cos(t)$ 和 $y(t) = \sin(t)$. 令 $h > 0$, 下面是计算 $x_k \approx x(kh)$ 和 $y_k \approx y(kh)$ 的三种迭代法, 设 $x_0 = 1$ 和 $y_k = 0$ :

方法 1: $x_{k+1} = 1 + hy_k$

$\quad\quad\quad y_{k+1} = 1 - hx_k$

方法 2: $x_{k+1} = 1 + hy_k$

$\quad\quad\quad y_{k+1} = 1 - hx_{k+1}$

方法 3: $x_{k+1} = 1 + hy_{k+1}$

$\quad\quad\quad y_{k+1} = 1 - hx_{k+1}$

用格式

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = A_h \begin{bmatrix} x_k \\ y_k \end{bmatrix}$$

来表达每种方法, 这里 $\boldsymbol{A}_h$ 是 $2 \times 2$ 矩阵. 对每一种情形计算 $\lambda(\boldsymbol{A}_h)$ 并用之讨论 $k \to \infty$ 时的 $\lim x_k$ 和 $\lim y_k$.

**7.1.11** 若 $\boldsymbol{J} \in \mathbb{R}^{d\times d}$ 是一个 Jordan 块, $\kappa_\infty(\boldsymbol{J})$ 是多少?

**7.1.12** 证明: 若

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ \boldsymbol{0} & \boldsymbol{R}_{22} \end{bmatrix} \begin{matrix} p \\ q \end{matrix}$$
$$\quad p \quad q$$

是正规矩阵且 $\lambda(\boldsymbol{R}_{11}) \cap \lambda(\boldsymbol{R}_{22}) = \varnothing$, 则 $\boldsymbol{R}_{12} = 0$.

## 本节注释与参考文献

Wilkinson(1965, 第 1 章) 和 Stewart(1973, 第 6 章) 精美地介绍了代数特征值问题之数学性质, 对需要更多参阅文献的读者, 我们推荐:

R. Bellman (1970). *Introduction to Matrix Analysis,* 2nd ed., McGraw-Hill, New York.

I. C. Gohberg, P. Lancaster, and L. Rodman (1986). *Invariant Subspaces of Matrices With Applications,* John Wiley and Sons, New York.

M. Marcus and H.Minc (1964). *A. Survey of Matrix Theory and Matrix Inequalities,* Allyn and Bacon, Boston.

L. Mirsky (1963). *An Introduction to Linear Algebra* Oxford Universfty Press, Oxford.

Schur 分解最早发表于:

I. Schur(1909). "On the Characteristic Roots of a Linear Substitution with an Application to the Theory of Integral Equations." *Math. Ann.* 66, 488–510(German).

与我们的证明十分类似的一个证明可见如下专著的 105 页.

H. W. Turnbull and A.C. Aitken(1961). *An Introduction to the Theory of Canonical Forms*, Dover, New York.

有关奇异值、特征值、伪特征值 (见 11.3.4 节) 之间的关系之讨论可见:

K-C. Toh and L.N.Trefethen (1994). "Pseudozeros of Polynomials and Pseudospectra of Companion Matrices, " *Numer. Math.* 68, 403–425.

F. Kittaneh(1995). "Singular Values of Companion Matrices and Bounds on zeros of Polynomials," *SIAM J. Matrix Anal. Appl.* 16, 333–340.

## 7.2 扰动理论

计算特征值也就是计算特征多项式的零点. Galois 理论告诉我们, 如果 $n > 4$, 这样的过程必须是迭代的, 这样, 有限终止必带来误差. 为了得到合理的迭代终止标准, 我们需要扰动理论来告诉我们怎样考虑近似特征值和不变子空间.

### 7.2.1 特征值的敏感度

几个求特征值程序产生一列相似变换 $\boldsymbol{X}_k$, 使得 $\boldsymbol{X}_k^{-1}\boldsymbol{A}\boldsymbol{X}_k$ 逐步地 "更对角化". 自然会问, 矩阵的对角元素与它的特征值之间的相似程度怎样?

**定理 7.2.1(Gershgorin 圆盘定理)** 如果 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} = \boldsymbol{D}+\boldsymbol{F}$, 其中 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n)$ 且 $F$ 的对角元素为零, 那么

$$\lambda(\boldsymbol{A}) \subseteq \bigcup_{i=1}^{n} D_i,$$

其中$D_i = \left\{z_i \in \mathbb{C} : |z-d_i| \leqslant \sum_{j=1}^{n} |f_{ij}|\right\}$.

**证明** 假设 $\lambda \in \lambda(\boldsymbol{A})$ 且不失一般性, $\lambda \neq d_i$, $i = 1:n$. 由于 $(\boldsymbol{D}-\lambda\boldsymbol{I})+\boldsymbol{F}$ 奇异, 由引理 2.3.3 得,

$$1 \leqslant \|(\boldsymbol{D}-\lambda\boldsymbol{I})^{-1}\boldsymbol{F}\|_\infty = \sum_{j=1}^{n} \frac{|f_{kj}|}{|d_k-\lambda|}$$

对某一 $k(1 \leqslant k \leqslant n)$ 成立. 但这隐含 $\lambda \in D_k$. □

还可证明, 如果 Gershgorin 圆盘 $D_i$ 与其他圆盘孤立, 那么 $D_i$ 必定恰好包含 $\boldsymbol{A}$ 的一个特征值. 参看 Wilkinson(1965, 从 71 页起).

**例 7.2.1** 如果

$$A = \begin{bmatrix} 10 & 2 & 3 \\ -1 & 0 & 2 \\ 1 & -2 & 1 \end{bmatrix},$$

那么 $\lambda(A) \approx \{10.226, 0.3870 + 2.2216\mathrm{i}, 0.3870 - 2.2216\mathrm{i}\}$ 且 Gershgorin 圆盘是

$$D_1 = \{|z| : |z-10| \leqslant 5\},\ D_2 = \{|z| : |z| \leqslant 3,\},D_3 = \{|z| : |z-1| \leqslant 3\}.$$

对于一些很重要的求特征值程序, 能够证明算出的特征值是矩阵 $A+E$ 的精确特征值, 这里 $E$ 的范数很小. 所以, 我们必须了解一个矩阵的特征值是如何受微小扰动所影响的. 阐明这个问题的典型结果是下面的定理.

**定理 7.2.2(Bauer-Fike)** 如果 $\mu$ 是 $A+E \in \mathbb{C}^{n\times n}$ 的一个特征值且 $X^{-1}AX = D = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$, 则

$$\min_{\lambda\in\lambda(A)} |\lambda - \mu| \leqslant \kappa_p(X)||E||_p,$$

其中 $||\cdot||_p$ 表示任一 $p$ 范数.

**证明** 我们只需考虑 $\mu$ 不在 $\lambda(A)$ 中的情形. 如果矩阵 $X^{-1}(A+E-\mu I)X$ 奇异, 那么 $I + (D-\mu I)^{-1}(X^{-1}EX)$ 也奇异. 于是从引理 2.3.3, 我们有

$$\begin{aligned} 1 &\leqslant ||(D-\mu I)^{-1}(X^{-1}EX)||_p \\ &\leqslant ||(D-\mu I)^{-1}||_p||X||_p||E||_p||X^{-1}||_p. \end{aligned}$$

由于 $(D-\mu I)^{-1}$ 是对角矩阵, 而对角矩阵的 $p$ 范数是最大对角元素的绝对值, 故 $||(D-\mu I)^{-1}||_p = \min\limits_{\lambda\in\lambda(A)} \dfrac{1}{|\lambda-\mu|}$. 由此, 可知定理成立. □

通过 Schur 分解可以得到一个类似结果.

**定理 7.2.3** 设 $Q^{\mathrm{H}}AQ = D+N$ 是 $A \in \mathbb{C}^{n\times n}$ 的一个 Schur 分解 (见 (7.1.3)). 如果 $\mu \in \lambda(A+E)$ 且 $p$ 是使得 $|N|^p = 0$ 成立的最小正整数, 则

$$\min_{\lambda\in\lambda(A)} |\lambda-\mu| \leqslant \max(\theta, \theta^{1/p}),$$

其中

$$\theta = ||E||_2 \sum_{k=0}^{p-1} ||N||_2^k.$$

**证明** 定义

$$\delta = \min_{\lambda\in\lambda(A)} |\lambda-\mu| = \frac{1}{||(\mu I - D^{-1})||_2}.$$

当 $\delta = 0$ 时定理显然成立. 若 $\delta > 0$, 那么 $I - (\mu I - A)^{-1}E$ 奇异且由引理 2.3.3 我们有

$$1 \leqslant ||(\mu I - A)^{-1}E||_2 \leqslant ||(\mu I - A)^{-1}||_2||E||_2$$

$$=\|((\mu \boldsymbol{I}-\boldsymbol{D})-\boldsymbol{N})^{-1}\|_2\|\boldsymbol{E}\|_2. \tag{7.2.1}$$

由于 $(\mu \boldsymbol{I}-\boldsymbol{D})^{-1}$ 为对角矩阵且 $|\boldsymbol{N}|^p=0$, 不难证明 $((\mu \boldsymbol{I}-\boldsymbol{D})^{-1}\boldsymbol{N})^p=0$. 这样

$$((\mu \boldsymbol{I}-\boldsymbol{D})-\boldsymbol{N})^{-1}=\sum_{k=0}^{p-1}((\mu \boldsymbol{I}-\boldsymbol{D})^{-1}\boldsymbol{N})^k(\mu \boldsymbol{I}-\boldsymbol{D})^{-1},$$

因此,

$$\|((\mu \boldsymbol{I}-\boldsymbol{D})-\boldsymbol{N})^{-1}\|_2 \leqslant \frac{1}{\delta}\sum_{k=0}^{p-1}\left(\frac{\|\boldsymbol{N}\|_2}{\delta}\right)^k.$$

若 $\delta>1$, 那么

$$\|(\mu \boldsymbol{I}-\boldsymbol{D})-N)^{-1}\|_2 \leqslant \frac{1}{\delta}\sum_{k=0}^{p-1}\|\boldsymbol{N}\|_2^k,$$

且从 (7.2.1) 知, $\delta \leqslant \theta$. 若 $\delta \leqslant 1$, 则

$$\|(\mu \boldsymbol{I}-\boldsymbol{D})-\boldsymbol{N})^{-1}\|_2 \leqslant \frac{1}{\delta^p}\sum_{k=0}^{p-1}\|\boldsymbol{N}\|_2^k,$$

且从 (7.2.1) 知, $\delta^p \leqslant \theta$. 所以, $\delta \leqslant \max(\theta, \theta^{1/p})$. □

**例 7.2.2** 如果

$$\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\ 0 & 4 & 5\\ 0 & 0 & 4.001\end{bmatrix} \quad 且 \quad \boldsymbol{E}=\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0.001 & 0 & 0\end{bmatrix},$$

那么 $\lambda(\boldsymbol{A}+\boldsymbol{E}) \approx \{1.0001, 4.0582, 3.9427\}$ 且 $\boldsymbol{A}$ 的特征向量矩阵满足 $\kappa_2(\boldsymbol{X}) \approx 10^7$. 定理 7.2.2 中的 Bauer-Fike 界的数量级为 $10^4$ 而定理 7.2.3 中的 Schur 界的数量级为 $10^0$.

若 $\boldsymbol{A}$ 非正规, 定理 7.2.2 和定理 7.2.3 都表明了潜在的特征值灵敏度. 特别是当 $\kappa_2(X)$ 或 $\|\boldsymbol{N}\|_2^{p-1}$ 很大时, $\boldsymbol{A}$ 的微小变动就会导致特征值的很大变化.

**例 7.2.3** 如果

$$\boldsymbol{A}=\begin{bmatrix}0 & \boldsymbol{I}_9\\ 0 & 0\end{bmatrix} \quad 且 \quad \boldsymbol{E}=\begin{bmatrix}0 & 0\\ 10^{-10} & 0\end{bmatrix},$$

那么对于所有 $\lambda \in \lambda(\boldsymbol{A})$ 且 $\mu \in \lambda(\boldsymbol{A}+\boldsymbol{E})$, $|\lambda-\mu|=10^{-1}$. 在本例中, 矩阵 $\boldsymbol{A}$ 量级为 $10^{-10}$ 的变化导致了其特征值量级为 $10^{-1}$ 的变化.

### 7.2.2 单特征值的条件数

如果矩阵 $\boldsymbol{A}$ 是正规的, 那么它的特征值是不会极端敏感的. 另一方面, 非正规性并不一定隐含其特征值很灵敏. 其实, 一个非正规矩阵可能既有良态的特征值又有病态的特征值. 基于这个理由, 有必要完善我们的扰动理论以便将之应用于单个特征值而不是整个谱.

为此, 假设 $\lambda$ 是 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 的单特征值且 $\boldsymbol{x},\boldsymbol{y}$ 满足 $\boldsymbol{Ax}=\lambda\boldsymbol{x}$, $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{A}=\lambda\boldsymbol{y}^{\mathrm{H}}$, $\|\boldsymbol{x}\|_2=\|\boldsymbol{y}\|_2=1$. 如果 $\boldsymbol{Y}^{\mathrm{H}}\boldsymbol{AX}=\boldsymbol{J}$ 是 Jordan 分解, $\boldsymbol{Y}^{\mathrm{H}}=\boldsymbol{X}^{-1}$, 那么 $\boldsymbol{y}$ 和 $\boldsymbol{x}$ 是 $X(:,i)$ 和 $Y(:,i)$ 的非零倍数 (对某个 $i$). 从 $1=Y(:,i)^{\mathrm{H}}X(:,i)$ 可知 $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}\neq 0$, 该事实我们稍后要用到.

运用函数理论的经典结论, 可以证明: 在原点附近存在一个可微的 $x(\varepsilon)$ 和 $\lambda(\varepsilon)$, 使得

$$(\boldsymbol{A}+\varepsilon\boldsymbol{F})x(\varepsilon)=\lambda(\varepsilon)x(\varepsilon),\qquad \|\boldsymbol{F}\|_2=1,$$

其中 $\lambda(0)=\lambda$, $x(0)=x$. 对 $\varepsilon$ 微分该方程且令 $\varepsilon=0$, 我们得到

$$\boldsymbol{A}\dot{x}(0)+\boldsymbol{F}x=\dot{\lambda}(0)x+\lambda\dot{x}(0),$$

用 $y^{\mathrm{H}}$ 乘等式两边, 同时除以 $\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}$, 且取绝对值得

$$|\dot{\lambda}(0)|=\left|\frac{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{F}\boldsymbol{x}}{\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}}\right|\leqslant\frac{1}{|\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}|}.$$

若 $\boldsymbol{F}=\boldsymbol{y}\boldsymbol{x}^{\mathrm{H}}$, 即知上界可以达到. 基于此, 我们称 $s(\lambda)=|\boldsymbol{y}^{\mathrm{H}}\boldsymbol{x}|$ 的倒数为**特征值$\lambda$的条件数**.

粗略地说, 以上分析表明, 如果对矩阵 $\boldsymbol{A}$ 作量级为 $\varepsilon$ 的扰动, 那么其特征值 $\lambda$ 之扰动可能达到 $\varepsilon/s(\lambda)$. 这样, 如果 $s(\lambda)$ 小, 就有理由认为 $\lambda$ 是病态的. 注意到 $s(\lambda)$ 是与 $\lambda$ 相应的左右特征向量之夹角的余弦, 且只要 $\lambda$ 是单特征值则 $s(\lambda)$ 值就唯一.

$s(\lambda)$ 小意味着 $\boldsymbol{A}$ 接近一个有重特征值的矩阵. 特别地, 如果 $\lambda$ 是单特征值且 $s(\lambda)<1$, 那么存在一个 $\boldsymbol{E}$ 使得 $\lambda$ 是 $\boldsymbol{A}+\boldsymbol{E}$ 的重特征且

$$\frac{\|\boldsymbol{E}\|_2}{\|\boldsymbol{A}\|_2}\leqslant\frac{s(\lambda)}{\sqrt{1-s(\lambda)^2}}.$$

该结论是 Wilkinson(1972) 证明的.

**例 7.2.4** 如果

$$\boldsymbol{A}=\begin{bmatrix}1&2&3\\0&4&5\\0&0&4.001\end{bmatrix}\quad 和\quad \boldsymbol{E}=\begin{bmatrix}0&0&0\\0&0&0\\0.001&0&0\end{bmatrix},$$

那么 $\lambda(\boldsymbol{A}+\boldsymbol{E})\approx\{1.0001,4.0582,3.9427\}$ 且 $s(1)\approx 0.8\times10^{0}$, $s(4)\approx 0.2\times10^{-3}$, 且 $s(4.001)\approx 0.2\times10^{-3}$. 可以观察到 $\|\boldsymbol{E}\|_2/s(\lambda)$ 是对每个特征值所受之扰动的很好估计.

### 7.2.3 多重特征值的敏感度

如果 $\lambda$ 是多重特征值, 那么特征值的敏感度问题更复杂. 例如, 如果

$$\boldsymbol{A}=\begin{bmatrix}1&a\\0&1\end{bmatrix}且\boldsymbol{F}=\begin{bmatrix}0&0\\1&0\end{bmatrix},$$

那么 $\lambda(\boldsymbol{A}+\varepsilon\boldsymbol{F})=\{1\pm\sqrt{\varepsilon a}\}$. 注意到, 若 $a\neq 0$, 那么可以推出 $\boldsymbol{A}+\varepsilon\boldsymbol{F}$ 的特征值在零处不可微. 它们在原点的变化率是无穷. 一般地, 若 $\lambda$ 是 $\boldsymbol{A}$ 之一退化特征值, 那么 $\boldsymbol{A}$ 的 $O(\varepsilon)$ 扰动将导致 $\lambda$ 的 $O(\varepsilon^{1/p})$ 扰动 (若 $\lambda$ 对应一个 $p\times p$ Jordan 块). 详细讨论可见 Wilkinson(1965, 从 77 页起).

### 7.2.4　不变子空间的敏感度

敏感的特征向量集合可以组成一个非敏感的不变子空间, 只要此空间相应的密集的特征值是孤立的. 更精确地说, 假设

$$\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\begin{array}{c}\left[\begin{array}{cc}\boldsymbol{T}_{11} & \boldsymbol{T}_{12}\\ \boldsymbol{0} & \boldsymbol{T}_{22}\end{array}\right]\begin{array}{c}r\\ n-r\end{array}\\ \begin{array}{cc}r & n-r\end{array}\qquad\quad\end{array}\tag{7.2.2}$$

是 $\boldsymbol{A}$ 的一个 Schur 分解, 其中

$$\boldsymbol{Q}=\underset{\quad r\qquad\quad n-r}{[\,\boldsymbol{Q}_1\qquad \boldsymbol{Q}_2\,]}.\tag{7.2.3}$$

由我们对特征向量扰动问题的讨论, 易知不变子空间 $\operatorname{ran}(\boldsymbol{Q}_1)$ 的敏感度依赖于 $\lambda(\boldsymbol{T}_{11})$ 和 $\lambda(\boldsymbol{T}_{22})$ 之间的距离. 该距离的适当度量正好是线性变换 $\boldsymbol{X}\to\boldsymbol{T}_{11}\boldsymbol{X}-\boldsymbol{X}\boldsymbol{T}_{22}$ 的最小奇异值.(这个变换曾在引理 7.1.5 中出现过.) 特别地, 如果我们定义矩阵 $\boldsymbol{T}_{11}$ 和 $\boldsymbol{T}_{22}$ 之间的分离度为:

$$\operatorname{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})=\min_{\boldsymbol{X}\neq\boldsymbol{0}}\frac{\|\boldsymbol{T}_{11}\boldsymbol{X}-\boldsymbol{X}\boldsymbol{T}_{22}\|_F}{\|\boldsymbol{X}\|_F}.\tag{7.2.4}$$

则我们有下面一般结论.

**定理 7.2.4**　*假设 (7.2.2) 和 (7.2.3) 成立且对任意矩阵 $\boldsymbol{E}\in\mathbb{C}^{n\times n}$, 我们划分 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{E}\boldsymbol{Q}$ 如下:*

$$\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{E}\boldsymbol{Q}=\begin{array}{c}\left[\begin{array}{cc}\boldsymbol{E}_{11} & \boldsymbol{E}_{12}\\ \boldsymbol{E}_{21} & \boldsymbol{E}_{22}\end{array}\right]\begin{array}{c}r\\ n-r\end{array}\\ \begin{array}{cc}r & n-r\end{array}\qquad\quad\end{array}$$

*如果 $\operatorname{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})>0$ 且*

$$\|\boldsymbol{E}\|_2\left(1+\frac{5\|\boldsymbol{T}_{12}\|_2}{\operatorname{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})}\right)\leqslant\frac{\operatorname{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})}{5},$$

*那么存在一个 $\boldsymbol{P}\in\mathbb{C}^{(n-r)\times r}$ 并且*

$$\|\boldsymbol{P}\|_2\leqslant 4\frac{\|\boldsymbol{E}_{21}\|_2}{\operatorname{sep}(T_{11},T_{22})}$$

*使得 $\hat{\boldsymbol{Q}}_1=(\boldsymbol{Q}_1+\boldsymbol{Q}_2\boldsymbol{P})(\boldsymbol{I}+\boldsymbol{P}^{\mathrm{H}}\boldsymbol{P})^{-1/2}$ 的各列组成 $\boldsymbol{A}+\boldsymbol{E}$ 不变子空间的标准正交基.*

**证明** 这个结论是 Stewart(1973) 之定理 4.11 的稍微变化, 欲知证明细节请参阅原作. 也可参看 Stewart and Sun(1990, 第 230 页). 矩阵 $(\boldsymbol{I}+\boldsymbol{P}^{\mathrm{H}}\boldsymbol{P})^{-1/2}$ 是对称正定矩阵 $\boldsymbol{I}+\boldsymbol{P}^{\mathrm{H}}\boldsymbol{P}$ 之平方根的逆矩阵. 参看 4.2.10 节. □

**推论 7.2.5** 如果定理 7.2.4 中假设成立, 那么

$$\mathrm{dist}(\mathrm{ran}(\boldsymbol{Q}_1),\mathrm{ran}(\hat{\boldsymbol{Q}}_1))\leqslant 4\frac{\|E_{21}\|_2}{\mathrm{sep}(T_{11},T_{22})}.$$

**证明** 利用 $\boldsymbol{P}$ 的奇异值分解, 可证

$$\|\boldsymbol{P}(\boldsymbol{I}+\boldsymbol{P}^{\mathrm{H}}\boldsymbol{P})^{-1/2}\|_2\leqslant\|\boldsymbol{P}\|_2 \tag{7.2.5}$$

因为所求之距离就是 $\boldsymbol{Q}_2^{\mathrm{H}}\hat{\boldsymbol{Q}}_1=\boldsymbol{P}(\boldsymbol{I}+\boldsymbol{P}^{\mathrm{H}}\boldsymbol{P})^{-1/2}$ 的范数, 推论得证.

这样, $\mathrm{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})$ 的倒数可以看作一个条件数, 用来度量不变子空间 $\mathrm{ran}(\boldsymbol{Q}_1)$ 的敏感度.

**例 7.2.5** 设

$$\boldsymbol{T}_{11}=\begin{bmatrix}3&10\\0&1\end{bmatrix},\boldsymbol{T}_{22}=\begin{bmatrix}0&-20\\0&3.01\end{bmatrix},\ \boldsymbol{T}_{12}=\begin{bmatrix}1&-1\\-1&1\end{bmatrix}.$$

还假定 $\boldsymbol{A}=\boldsymbol{T}=\begin{bmatrix}\boldsymbol{T}_{11}&\boldsymbol{T}_{12}\\\boldsymbol{0}&\boldsymbol{T}_{22}\end{bmatrix}$.

注意到 $\boldsymbol{A}\boldsymbol{Q}_1=\boldsymbol{Q}_1\boldsymbol{T}_{11}$, 这里 $\boldsymbol{Q}_1=[e_1,e_2]\in\mathbb{R}^{4\times2}$. 由计算知 $\mathrm{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})\approx 0.0003$. 若

$$\boldsymbol{E}_{21}=10^{-6}\begin{bmatrix}1&1\\1&1\end{bmatrix}$$

且我们检查

$$\boldsymbol{A}+\boldsymbol{E}=\begin{bmatrix}\boldsymbol{T}_{11}&\boldsymbol{T}_{12}\\\boldsymbol{E}_{21}&\boldsymbol{T}_{22}\end{bmatrix}$$

的 Schur 分解, 则发现 $\boldsymbol{Q}_1$ 受扰动变为

$$\hat{\boldsymbol{Q}}_1=\begin{bmatrix}-0.9999&-0.0003\\0.0003&-0.9999\\-0.0005&-0.0026\\0.0000&0.0003\end{bmatrix}.$$

于是, 我们有

$$\mathrm{dist}(\mathrm{ran}(\hat{\boldsymbol{Q}}_1),(\mathrm{ran}\boldsymbol{Q}_1))\approx 0.0027\approx 10^{-6}/\mathrm{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22}).$$

### 7.2.5 特征向量的敏感度

如果在上一节中令 $r=1$, 那么就是分析特征向量的敏感度问题.

**推论 7.2.6** 假设 $\boldsymbol{A},\boldsymbol{E}\in\mathbb{C}^{n\times n}$ 且 $\boldsymbol{Q}=[q_1\quad \boldsymbol{Q}_2]\in\mathbb{C}^{n\times n}$ 是酉矩阵, 其中 $q_1\in\mathbb{C}^n$. 设

$$Q^{\mathrm{H}}AQ=\begin{bmatrix}\lambda & v^{\mathrm{H}}\\ 0 & T_{22}\end{bmatrix}\begin{matrix}1\\ n-1\end{matrix}\quad Q^{\mathrm{H}}EQ=\begin{bmatrix}\varepsilon & r^{\mathrm{H}}\\ \delta & E_{22}\end{bmatrix}\begin{matrix}1\\ n-1\end{matrix}$$
$$\qquad 1\quad n-1 \qquad\qquad\qquad 1\quad n-1$$

(这样, $q_1$ 是特征向量.) 如果 $\sigma=\sigma_{\min}(T_{22}-\lambda I)>0$ 且

$$\|E\|_2\left(1+\frac{5\|v\|_2}{\sigma}\right)\leqslant\frac{\delta}{5},$$

那么存在满足 $\|p\|_2\leqslant 4\frac{\|\delta\|_2}{\sigma}$ 的 $p\in\mathbb{C}^{n-1}$, 使得 $\hat{q}_1=(q_1+Q_2p)/\sqrt{1+p^{\mathrm{H}}p}$ 是 $A+E$ 的单位 2 范数特征向量, 而且有

$$\mathrm{dist}(\mathrm{span}\{q_1\},\mathrm{span}\{\hat{q}_1\})\leqslant 4\frac{\|\delta\|_2}{\sigma}.$$

**证明** 从定理 7.2.4 和推论 7.2.5, 并注意到若 $T_{11}=\lambda$ 则有 $\mathrm{sep}(T_{11},T_{22})=\sigma_{\min}(T_{22}-\lambda I)$, 就可得此结论. □

$\sigma_{\min}(T_{22}-\lambda I)$ 只是粗略地估计 $\lambda$ 和 $T_{22}$ 的特征值之间的分离度. 之所以说“粗略”, 是因为

$$\mathrm{sep}(\lambda,T_{22})=\sigma_{\min}(T_{22}-\lambda I)\leqslant\min_{\mu\in\lambda(T_{22})}|\mu-\lambda|,$$

而此上界是一个粗略的估计.

特征值的分离度影响特征向量的敏感度是毫不奇怪的. 实际上, 若 $\lambda$ 为非退化的重特征值, 那么对应的不变子空间就有无穷多可能的特征向量基. 以上分析仅仅表明这种不确定性在特征值集结时出现. 换句话说. 相近的特征值所对应的特征向量是“不稳定的”.

**例 7.2.6** 如果

$$A=\begin{bmatrix}1.01 & 0.01\\ 0.00 & 0.99\end{bmatrix},$$

那么特征值 $\lambda=0.99$ 有条件数 $1/s(0.99)\approx1.118$ 且相应的特征向量 $x=[0.4472,-0.8944]^{\mathrm{T}}$. 另一方面, $A$ 的“附近的”矩阵

$$A+E=\begin{bmatrix}1.01 & 0.01\\ 0.00 & 1.00\end{bmatrix}$$

的特征值 $\hat{\lambda}=1.00$ 有特征向量

$$\hat{x}=(0.7071,-0.7071)^{\mathrm{T}}.$$

## 习 题

**7.2.1** 设 $Q^{\mathrm{H}}AQ=\mathrm{diag}(\lambda_1)+N$ 是 $A\in\mathbb{C}^{n\times n}$ 的一个 Schur 分解且定义 $\nu(A)=\|A^{\mathrm{H}}A-AA^{\mathrm{H}}\|_F$. 则有

$$\frac{\nu(A)^2}{6\|A\|_F^2}\leqslant\|N\|_F^2\leqslant\sqrt{\frac{n^3-n}{12}}\nu(A),$$

其上下界分别由 Henrici(1962) 和 Eberlein(1965) 给出. 当 $n=2$ 时, 试证明这些结果.

**7.2.2** 设 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 和 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}=\text{diag}(\lambda_1,\cdots,\lambda_n)$, 其中 $\lambda_i$ 互不相同. 证明: 若 $\boldsymbol{X}$ 的列向量是 2 范数下的单位向量, 则 $\kappa_F(\boldsymbol{X}^2)=n\sum\limits_{i=1}^{n}(1/s(\lambda_i))^2$.

**7.2.3** 设 $Q^{\text{H}}\boldsymbol{A}\boldsymbol{Q}=\text{diag}(\lambda_i)+\boldsymbol{N}$ 是 $\boldsymbol{A}$ 的 Schur 分解且 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}=\text{diag}(\lambda_i)$. 证明 $\kappa_2(\boldsymbol{X})^2\geqslant 1+(\|\boldsymbol{N}\|_F/\|\boldsymbol{A}\|_F)^2$. 见 Loizou(1969).

**7.2.4** 若 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}=\text{diag}(\lambda_i)$ 且 $|\lambda_1|\geqslant\cdots\geqslant|\lambda_n|$, 则

$$\frac{\sigma_i(\boldsymbol{A})}{\kappa_2(\boldsymbol{X})}\leqslant|\lambda_i|\leqslant\kappa_2(\boldsymbol{X})\sigma_i(\boldsymbol{A}).$$

在 $n=2$ 情形下证明这个结果. 见 Ruhe(1975).

**7.2.5** 证明: 若 $\boldsymbol{A}=\begin{bmatrix} a & c\\ 0 & b\end{bmatrix}$ 且 $a\neq b$, 则 $s(a)=s(b)=(1+|c/(a-b)|^2)^{-1/2}$.

**7.2.6** 假设

$$\boldsymbol{A}=\begin{bmatrix}\lambda & \boldsymbol{v}^{\text{T}}\\ \boldsymbol{0} & \boldsymbol{T}_{22}\end{bmatrix}$$

且 $\lambda\notin\lambda(\boldsymbol{T}_{22})$. 证明: 若 $\sigma=\text{sep}(\lambda,\boldsymbol{T}_{22})$, 则

$$s(\lambda)=\frac{1}{\sqrt{1+\|(\boldsymbol{T}_{22}-\lambda\boldsymbol{I})^{-1}\boldsymbol{v}\|_2^2}}\leqslant\frac{\sigma}{\sqrt{\sigma^2+\|\boldsymbol{v}\|_2^2}}$$

**7.2.7** 证明单特征值的条件数在酉相似变换下不变.

**7.2.8** 在 Bauer-Fike 定理 (定理 7.2.2) 中的同样假设下, 证明

$$\min_{\lambda\in\lambda(\boldsymbol{A})}|\lambda-\mu|\leqslant\|\,|\boldsymbol{X}^{-1}|\boldsymbol{E}||\boldsymbol{X}|\,\|_p.$$

**7.2.9** 证明 (7.2.5)

**7.2.10** 证明: 如果 $\boldsymbol{B}\in\mathbb{C}^{m\times m}$ 且 $\boldsymbol{C}\in\mathbb{C}^{n\times n}$, 则对所有 $\lambda\in\lambda(\boldsymbol{B})$ 和 $\mu\in\lambda(\boldsymbol{C})$ 有 $\text{sep}(\boldsymbol{B},\boldsymbol{C})\leqslant|\lambda-\mu|$.

## 本节注释与参考文献

本节的许多结果出现在 Wikinson(1965, 第 2 章) 和 Stewart and Sun(1990) 中, 也出现在下列文献中:

F. L. Bauer and C. T. Fike (1960), "Norms and Exclusion Theorems," *Numer, Math.* 2, 123–144.

A. S. Householder(1964). *The Theory of Matrices in Numerical Analysis.* Blaisdell, New York.

下列文章分析了扰动对广义矩阵的特征值影响:

A. Ruhe(1970). "Perturbation Bounds for Means of Eigenvalues and Invariant Subspaces," *BIT* 10, 343–354.

A. Ruhe(1970). "Properties of a Matrix with a Very Ill-Conditioned Eigenproblem," *Numer. Math.* 15, 57–60.

J. H. Wilkinson(1972). "Note on Matrices with a Very Ill-Conditioned Eigenproblem," *Numer. Math.* 19, 176–178.

W. Kahan, B. N. Parlett, and E. Jiang(1982). "Residual Bounds on Approximate Eigensystems of Nonnormal Matrices, " *SLAM J. Numer. Anal.* 19, 470–484.

J. H. Wilkinson(1984). "On Neighboring Matrices with Quadratic Elementary Divisors," *Numer, Math.* 44, 1–21.

J. V. Burke and M. L. Overton(1992). "Stable Perturbations of Nonsymmetric Matrices," *Lin. Alg. and Its Application* 171, 249–273.

Wilkinson 关于几乎亏损矩阵的工作是在大量涌现的有关 "几乎亏损" 问题的文献中最典型的. 见:

N. J. Higham(1985). "Nearness Problems in Numerical Linear Algebra, " PhD Thesis, University of Manchester, England.

C. Van Loan(1985). "How Near is a Stable Matrix to an Unstable Matrix?, " *Contem-porary Mathematics, Vol.* 47, 465–477.

J. W. Demmel(1987). "On the Distance to the Nearest Ill-Posed Problem, " *Numer. Math.* 51, 251–289.

J. W. Demmel(1987). "A Counterexample for two Conjectures About Stability," *IEEE Trans, Auto. Cont. AC*-32, 340–342.

A. Ruhe(1987). "Closest Normal Matrix Found!," *BIT* 27, 585–598.

R. Byers(1988). "A Bisection Method for Measuring the Distance of a Stable Matrix to the Unstable Matrices, " *SIAM J. Sci. and Stat. Comp.* 9, 975 881.

J. W. Demmel(1988). "The Probability that a Numerical Analysis Problem is Difficult," *Math. Comp.* 50, 449–480.

N. J. Higham (1989). "Matrix Nearness Problems and Applicatons, " in *Applications of Matrix Theory,* M. J. C. Gover and S. Barnett(eds,) Oxford University Press, Oxford UK, 1–27

特征值条件的性质可参见:

C. Van Loan(1987). "On Estimating the Condition of Eigenvalues and Eigenvectors," *Lin. Alg. and Its Applic.* 88/89, 715–732.

C. D. Meyer and G. W. Stewart (1988). "Derivatives and Perturbations of Eigenvectors," *SIA M J. Num. Anal.* 25, 679–691.

G. W. Stewart and G. Zhang(1991). "Eigenvalues of Graded Matrices and the Condition Numbers of Multiple Eigenvalues, " *Numer. Math.* 58, 703–712.

J. G. Sun(1992). "On Condition Numbers of a Nondefective Multiple Eigenvalue," *Numer. Math.* 61, 265–276.

特征值条件数、正规性的偏离度和特征向量矩阵的条件之间的关系在下述文献中谈到:

P. Henrici(1962). "Bounds for Iterates, Inverses, spectial Variation and Fields of Values of Non-normal Matrices," *Numer. Math.* 4, 24–40.

P. Eberlein(1965). "On Measures of Non-Normality for Matrices," *Amer. math. Soc. Monthly* 72, 995–996.

R. A. Smith(1967). "The Condition Numbers of the Matrix Eigenvalue Problem," *Numer.Math.* 10, 232–240.

G. Loizou(1969). "Nonnormality and Jordan Condition Numbers of Matrices," *J. ACM* 16, 580–540.

A. van der Sluis(1975). "Perturbations of Eigenvalues of Non-normal Matrices," *Comm. ACM* 18, 30–36.

P. Henrici(1962) 也包含一个与定理 7.2.3 类似的结论. 下列文献深刻分析了不变子空间扰动:

T. Kato(1966).*Perturbation Theory for Linear Operators*, Sprigner-Verllag, New York.

C. Davis and W. M. Kahan(1970). "The Rotation of Eigenvectors by a Perturbation, III," *SIAM J. Num. Anal.* 7, 1–46.

G. W. Stewart(1971). "Error Bounds for Approximate Invariant Subspaces of Closed Linear Operators," *SIAM. J. Num. Anal.* 8, 796–808.

G. W. Stewart(1973). "Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems," *SIAM Review* 15, 727–764.

关于函数 sep(.,.) 和映射 $\boldsymbol{X} \to \boldsymbol{AX} + \boldsymbol{XA}^{\mathrm{T}}$ 的详细分析在下述文献中给出.

J. Varah(1979). "On the separation of Two Matrices," *SIAM J. Num. Anal.* 16, 216–222.

R. Byers and S. G. Nash(1987). "On the Singular Vectors of the Lyapunov Operator," *SIAM J. Alg. and Disc. Methods* 8, 59–66.

Gershgorin 定理能可用于推导复杂的扰动理论. 见 Wilkinson(1965, 第 2 章). 定理本身能用不同的方法加以概括和推广; 见:

R. S. Varga(1970). "Minimal Gershgorin Sets for Partitioned Matrices," *SIAM J. Num. Anal.* 7, 493–507.

R. J. Johnston(1971). "Gershgorin Theorems for Partitioned Matrices, " *Lin. Alg. and Its Ipplic.* 4, 205–220.

## 7.3 幂迭代法

假设给定 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ 和酉矩阵 $\boldsymbol{U}_0 \in \mathbb{C}^{n\times n}$. 假定 Householder 正交化 (算法 5.2.1) 能推广到复矩阵 (这能办到). 考虑如下迭代:

$$
\begin{aligned}
&T_0 = \boldsymbol{U}_0^{\mathrm{H}} \boldsymbol{A} \boldsymbol{U}_0 \\
&\textbf{for} \quad k = 1, 2, \cdots \\
&\qquad T_{k-1} = \boldsymbol{U}_k \boldsymbol{R}_k \quad (\text{QR分解}) \\
&\qquad T_k = \boldsymbol{R}_k \boldsymbol{U}_k \\
&\textbf{end}
\end{aligned} \tag{7.3.1}
$$

由于 $\boldsymbol{T}_k = \boldsymbol{R}_k \boldsymbol{U}_k = \boldsymbol{U}_k^{\mathrm{H}}(\boldsymbol{U}_k \boldsymbol{R}_k)\boldsymbol{U}_k = \boldsymbol{U}_k^{\mathrm{H}} \boldsymbol{T}_{k-1} \boldsymbol{U}_k$, 由归纳法可得

$$
\boldsymbol{T}_k = (\boldsymbol{U}_0 \boldsymbol{U}_1 \cdots \boldsymbol{U}_k)^{\mathrm{H}} \boldsymbol{A} (\boldsymbol{U}_0 \boldsymbol{U}_1 \cdots \boldsymbol{U}_k). \tag{7.3.2}
$$

这样, 每个 $\boldsymbol{T}_k$ 酉相似于 $\boldsymbol{A}$. $\boldsymbol{T}_k$ 几乎总是收敛到上三角形矩阵, 这正是显然的, 这正是本节的中心议题. 即 (7.3.2) 几乎总是 “收敛” 于 $\boldsymbol{A}$ 的 Schur 分解.

迭代 (7.3.1) 称为 QR迭代, 它构成计算 Schur 分解的最有效算法的支柱. 为了导出方法并得出其收敛性质, 首先给出本身也重要的两种别的求特征值的迭代法: 幂法和正交迭代法.

### 7.3.1 幂法

假设 $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ 可对角化且 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} = \operatorname{diag}(\lambda_1, \cdots, \lambda_n)$, 其中 $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, $|\lambda_1| > |\lambda_2| \geqslant \cdots \geqslant |\lambda_n|$. 给出一 2 范数单位向量 $\boldsymbol{q}^{(0)} \in \mathbb{C}^n$, 幂法产生如下向量序列 $\boldsymbol{q}^{(k)}$:

$$
\begin{aligned}
&\textbf{for} \quad k = 1, 2, \cdots \\
&\qquad z^{(k)} = Aq^{(k-1)} \\
&\qquad q^{(k)} = z^{(k)}/\|z^{(k)}\|_2 \\
&\qquad \lambda^{(k)} = [q^{(k)}]^{\mathrm{H}} A q^{(k)} \\
&\textbf{end}
\end{aligned} \tag{7.3.3}
$$

做 2 范数单位化并没有任何特别之处, 它只是让本节的讨论能较好地统一.

让我们考察幂迭代法的收敛性质. 如果

$$
q^{(0)} = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n
$$

且 $a_1 \neq 0$, 那么可以推出

$$
\boldsymbol{A}^k q^{(0)} = a_1 \lambda_1^k \left( x_1 + \sum_{j=2}^{n} \frac{a_j}{a_1} \left( \frac{\lambda_j}{\lambda_1} \right)^k x_j \right).
$$

由于 $q^{(k)} \in \operatorname{span}\{\boldsymbol{A}^k q^{(0)}\}$, 所以

$$
\operatorname{dist}(\operatorname{span}\{q^{(k)}\}, \operatorname{span}\{x_1\}) = O\left( \left| \frac{\lambda_2}{\lambda_1} \right|^k \right),
$$

而且

$$
|\lambda_1 - \lambda^{(k)}| = O\left( \left| \frac{\lambda_2}{\lambda_1} \right|^k \right).
$$

如果 $|\lambda_1| > |\lambda_2| \geqslant \cdots \geqslant |\lambda_n|$, 我们称 $\lambda_1$ 是主特征值. 这样, 如果 $\lambda_1$ 是主要的且 $q^{(0)}$ 在相应的主特征向量$x_1$ 方向上有一分量, 那么幂法收敛.

没有这些假定的迭代性质在 Wilkinson(1965, 570 页) 和 Parlett and Poole(1973) 中有讨论.

**例 7.3.1** 如果

$$A = \begin{bmatrix} -261 & 209 & -49 \\ -530 & 422 & -98 \\ -800 & 631 & -144 \end{bmatrix}$$

则 $\lambda(A) = \{10, 4, 3\}$, 在 (7.3.3) 中取 $q^{(0)} = [1, 0, 0]^{\mathrm{T}}$. 我们有

| $k$ | $\lambda^{(k)}$ |
|---|---|
| 1 | 13.0606 |
| 2 | 10.7191 |
| 3 | 10.2073 |
| 4 | 10.0633 |
| 5 | 10.0198 |
| 6 | 10.0063 |
| 7 | 10.0020 |
| 8 | 10.0007 |
| 9 | 10.0002 |

在实际中, 幂法的有用性取决于比值 $|\lambda_2|/|\lambda_1|$, 因为它反映收敛速度. 不用担心 $q^{(0)}$ 在 $x_1$ 方向上分量为零, 因为迭代过程中舍入误差通常能保证迭代序列 $q^{(k)}$ 在此方向上有分量. 而且, 常常在求主特征值和主特征向量的应用中已知道 $x_1$ 的预先估计. 一般地, 令 $q^0$ 即为此估计, 可以极小化出现 $a_1$ 很小的危险.

注意到应用幂法所需唯一之事是编制计算矩阵与向量相乘 $Aq$ 的子程序. 没必要用一个 $n \times n$ 的数组来存 $A$. 正由于此, 当 $A$ 是大型稀疏矩阵且 $|\lambda_1|$ 和 $|\lambda_2|$ 相差较大时, 此算法是很有效的.

用前一节提出的扰动理论能够获得 $|\lambda^{(k)} - \lambda_1|$ 的误差估计. 定义向量 $r^{(k)} = Aq^{(k)} - \lambda^{(k)}q^{(k)}$ 并注意到 $(A + E^{(k)})q^{(k)} = \lambda^{(k)}q^{(k)}$, 这里 $E^{(k)} = -r^{(k)}[q^{(k)}]^{\mathrm{H}}$. 这样 $\lambda^{(k)}$ 是 $A + E^{(k)}$ 的特征值且

$$|\lambda^{(k)} - \lambda_1| \approx \frac{\|E^{(k)}\|_2}{s(\lambda_1)} = \frac{\|r^{(k)}\|_2}{s(\lambda_1)}.$$

如果我们用幂法去产生近似的左、右主特征向量, 那么有可能获得 $s(\lambda_1)$ 的一个估计. 特别地, 如果 $w^{(k)}$ 是在 $(A^{\mathrm{H}})^k w^{(0)}$ 方向上的 2 范数单位向量, 那么我们作近似 $s(\lambda_1) \approx |w^{(k)\mathrm{H}}q^{(k)}|$.

### 7.3.2 正交迭代法

幂法的一个直接推广可用来计算高维的不变子空间. 令 $r$ 是一选定整数, 满足 $1 \leqslant r \leqslant n$. 给定具有正交列的 $n \times r$ 矩阵 $Q_0$, 正交迭代法产生如下的一列矩阵 $\{Q_k\} \subseteq \mathbb{C}^{n \times r}$:

$$\begin{aligned}&\textbf{for}\quad k=1,2,\cdots\\&\qquad Z_k=AQ_{k-1}\\&\qquad Q_kR_k=\boldsymbol{Z}_k\quad(\text{QR分解})\\&\textbf{end}\end{aligned}\tag{7.3.4}$$

注意到, 如果 $r=1$, 这就是幂法. 而且, 序列 $\{\boldsymbol{Q}_k e_1\}$ 正是幂法当初值为 $\boldsymbol{q}^{(0)}=\boldsymbol{Q}_0 e_1$ 时所产生的向量序列.

为了分析迭代的表现, 设

$$\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}=\mathrm{diag}(\lambda_i)+\boldsymbol{N},\ |\lambda_1|\geqslant|\lambda_2|\geqslant\cdots\geqslant|\lambda_n|\tag{7.3.5}$$

是 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 的 Schur 分解. 假设 $1\leqslant r<n$ 且 $\boldsymbol{Q},\boldsymbol{T}$ 和 $\boldsymbol{N}$ 划分如下:

$$\boldsymbol{Q}=\underset{\quad r\quad\ n-r}{\begin{bmatrix}\boldsymbol{Q}_\alpha & \boldsymbol{Q}_\beta\end{bmatrix}}\quad \boldsymbol{T}=\underset{\ r\quad\ n-r}{\begin{bmatrix}\boldsymbol{T}_{11} & \boldsymbol{T}_{12}\\ \boldsymbol{0} & \boldsymbol{T}_{22}\end{bmatrix}}\begin{matrix}r\\n-r\end{matrix}$$

$$\boldsymbol{N}=\underset{\ r\quad\ n-r}{\begin{bmatrix}\boldsymbol{N}_{11} & \boldsymbol{N}_{12}\\ \boldsymbol{0} & \boldsymbol{N}_{22}\end{bmatrix}}\begin{matrix}r\\n-r\end{matrix}.\tag{7.3.6}$$

若 $|\lambda_r|>|\lambda_{r+1}|$, 那么子空间 $D_r(\boldsymbol{A})=\mathrm{ran}(\boldsymbol{Q}_\alpha)$ 称为**主不变子空间**. 它是与特征值 $\lambda_1,\cdots,\lambda_r$ 所对应的唯一不变子空间. 下面的定理表明, 在合理假设下, 由 (7.3.4) 产生的子空间 $\mathrm{ran}(\boldsymbol{Q}_k)$ 以与 $|\lambda_{r+1}/\lambda_r|^k$ 成正比例的速度收敛到 $D_r(\boldsymbol{A})$.

**定理 7.3.1**　设 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 的 Schur 分解由 (7.3.5) 和 (7.3.6) 给定, $n\geqslant 2$. 假设 $|\lambda_r|>|\lambda_{r+1}|$ 且 $\theta\geqslant 0$ 满足

$$(1+\theta)|\lambda_r|>\|N\|_F.$$

如果 $\boldsymbol{Q}_0\in\mathbb{C}^{n\times r}$ 的列正交且

$$d=\mathrm{dist}(D_r(\boldsymbol{A}^{\mathrm{H}}),\mathrm{ran}(\boldsymbol{Q}_0))<1,$$

那么由 (7.3.4) 产生的矩阵 $\boldsymbol{Q}_k$ 满足

$$\mathrm{dist}(D_r(\boldsymbol{A}),\mathrm{ran}(\boldsymbol{Q}_k))\leqslant\frac{(1+\theta)^{n-2}}{\sqrt{1-d^2}}\left(1+\frac{\|\boldsymbol{T}_{12}\|_F}{\mathrm{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})}\right)\cdot\left(\frac{|\lambda_{r+1}|+\|\boldsymbol{N}\|_F/(1+\theta)}{|\lambda_r|-\|\boldsymbol{N}\|_F/(1+\theta)}\right)^k.$$

**证明**　此证明放在本节末的附录中.

定理 7.3.1 中的条件 $d<1$ 确保初始矩阵 $\boldsymbol{Q}$ 在某一特征方向上是非退化的:

$$d<1\leftrightarrow D_r(\boldsymbol{A}^{\mathrm{H}})^{\perp}\cap\mathrm{ran}(\boldsymbol{Q}_0)=\{0\}.$$

定理本质上表明, 如果这个条件满足且 $\theta$ 取的足够大, 则

$$\mathrm{dist}(D_r(\boldsymbol{A}),\mathrm{ran}(\boldsymbol{Q}_k))\leqslant c\left|\frac{\lambda_{r+1}}{\lambda_r}\right|^k$$

这里 $c$ 依赖 $\text{sep}(\boldsymbol{T}_{11},\boldsymbol{T}_{22})$ 以及 $\boldsymbol{A}$ 与正规性的偏离. 不用说, 如果 $|\lambda|$ 和 $|\lambda_{r+1}|$ 之间相差不够大, 收敛则会很慢.

**例 7.3.2** 如果把 (7.3.4) 用于例 7.3.1 中矩阵 $\boldsymbol{A}$, 这里 $\boldsymbol{Q}_0=[e_1,e_2]$, 我们发现:

| $k$ | $\text{dist}(D_2(\boldsymbol{A}),\,\text{ran}(\boldsymbol{Q}_k))$ |
|---|---|
| 1 | 0.0052 |
| 2 | 0.0047 |
| 3 | 0.0039 |
| 4 | 0.0030 |
| 5 | 0.0023 |
| 6 | 0.0017 |
| 7 | 0.0013 |

误差以 $(\lambda_3/\lambda_2)^k=(3/4)^k$ 趋向于零.

用 Stewart(1976) 的技巧可加速正交迭代的收敛速度. 在加速方案中, 近似特征值 $\lambda_i^{(k)}$ 满足

$$|\lambda_i^{(k)}-\lambda_i|\approx\left|\frac{\lambda_{r+1}}{\lambda_i}\right|^k,\quad i=1:r.$$

(没加速时右端是 $|\lambda_{i+1}/\lambda_i|^k$). Stewart 的算法常常涉及计算矩阵 $\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k$ 的 Schur 分解. 在 $\boldsymbol{A}$ 是大型稀疏矩阵且只需求它的几个最大特征值的情形下, 该方法是非常有用的.

### 7.3.3 QR 迭代法

我们现在来 "导出"(7.3.1) 中 QR 迭代并考察它的收敛性. 假设 (7.3.4) 中 $r=n$ 且 $\boldsymbol{A}$ 的特征值满足

$$|\lambda_1|>|\lambda_2|>\cdots>|\lambda_n|.$$

将 (7.3.5) 中的 $\boldsymbol{Q}$ 和 (7.3.4) 中的 $\boldsymbol{Q}_k$ 划分如下:

$$\boldsymbol{Q}=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_n],\quad \boldsymbol{Q}_n=[\boldsymbol{q}_1^{(k)},\cdots,\boldsymbol{q}_n^{(k)}].$$

如果

$$\text{dist}(D_i(\boldsymbol{A}^{\mathrm{H}}),\text{span}\{\boldsymbol{q}_1^{(0)},\cdots,\boldsymbol{q}_i^{(0)}\})<1,\quad i=1:n,\tag{7.3.7}$$

则从定理 7.3.1 可推出, 对 $i=1:n$ 有

$$\text{dist}(\text{span}\{\boldsymbol{q}_1^{(k)},\cdots,\boldsymbol{q}_i^{(k)}\},\text{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_i\})\to 0.$$

这说明由 $\boldsymbol{T}_k=\boldsymbol{Q}_k^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}_k$ 定义的矩阵 $\boldsymbol{T}_k$ 收敛到上三角形矩阵. 于是, 可以说, 只要初始迭代 $\boldsymbol{Q}_0\in\mathbb{C}^{n\times n}$ 不是退化矩阵 (满足 (7.3.7)), 正交迭代法就能算出 Schur 分解.

考虑怎样由前一个 $\boldsymbol{T}_{k-1}$ 直接算出矩阵 $\boldsymbol{T}_k$, 自然就产生了 QR 迭代. 一方面, 从 (7.3.4) 和 $\boldsymbol{T}_{k-1}$ 的定义中我们有

$$T_{k-1} = Q_{k-1}^{\mathrm{H}} A Q_{k-1} = Q_{k-1}^{\mathrm{H}}(AQ_{k-1}) = (Q_{k-1}^{\mathrm{H}} Q_k) R_k.$$

另一方面,

$$T_k = Q_k^{\mathrm{H}} A Q_k = (Q_k^{\mathrm{H}} A Q_{k-1})(Q_{k-1}^{\mathrm{H}} Q_k) = R_k (Q_{k-1}^{\mathrm{H}} Q_k)$$

这样, 先计算 $T_{k-1}$ 的 QR 分解, 然后将两个因子按逆序乘起来就决定了 $T_k$. 这也正是 (7.3.1) 中所做的.

**例 7.3.3** 如果迭代:

**for** $k = 1, 2, \cdots$
  $A = QR$
  $A = RQ$
**end**

应用到例 7.3.1 中的矩阵, 那么严格下三角元素按如下消去:

| $k$ | $O(\|a_{21}\|)$ | $O\|a_{31}\|$ | $O(\|a_{32}\|)$ |
|---|---|---|---|
| 1 | $10^{-1}$ | $10^{-1}$ | $10^{-2}$ |
| 2 | $10^{-2}$ | $10^{-2}$ | $10^{-3}$ |
| 3 | $10^{-2}$ | $10^{-3}$ | $10^{-3}$ |
| 4 | $10^{-3}$ | $10^{-3}$ | $10^{-3}$ |
| 5 | $10^{-3}$ | $10^{-4}$ | $10^{-3}$ |
| 6 | $10^{-4}$ | $10^{-5}$ | $10^{-3}$ |
| 7 | $10^{-4}$ | $10^{-5}$ | $10^{-3}$ |
| 8 | $10^{-5}$ | $10^{-6}$ | $10^{-4}$ |
| 9 | $10^{-5}$ | $10^{-7}$ | $10^{-4}$ |
| 10 | $10^{-6}$ | $10^{-8}$ | $10^{-4}$ |

注意到一步 QR 迭代的计算量是 $O(n^3)$. 而且, 既然收敛只是线性的 (若存在), 显然, 用该方法计算 Schur 分解的代价是相当昂贵的. 幸运的是, 这些实际困难能够克服, 见 7.4 节和 7.5 节.

### 7.3.4 LR 迭代法

本节我们简述基于 LU 分解而不是 QR 分解的幂迭代法. 令 $G_0 \in \mathbb{C}^{n\times r}$ 之秩为 $r$. 对应 (7.3.4) 我们有如下迭代:

**for** $k = 1, 2, \cdots$
  $Z_k = AG_{k-1}$
  $Z_k = G_k R_k$ (LU分解)   (7.3.8)
**end**

设 $r = n$ 且我们定义矩阵 $T_k$ 为

$$T_k = G_k^{-1} A G_k \tag{7.3.9}$$

可证明, 如果令 $L_0 = G_0$, 则 $T_k$ 可由如下方式产生:

$$
\begin{aligned}
&T_0 = L_0^{-1}AL_0 \\
&\textbf{for} \quad k = 1, 2, \cdots \\
&\qquad T_{k-1} = L_kR_k \qquad (\text{LU分解}) \qquad\qquad (7.3.10)\\
&\qquad T_k = R_kL_k \\
&\textbf{end}
\end{aligned}
$$

迭代 (7.3.8) 和 (7.3.10) 分别称为梯子迭代和 LR迭代. 在合理假设下, $\boldsymbol{T}_k$ 收敛于上三角形矩阵. 要成功地实现这两种方法, 必须结合选主元技巧. 参看 Wilkinson(1965, 602 页).

**附录**

为了建立定理 7.3.1, 我们需用到以下引理, 它关于矩阵及其逆的幂之界.

**引理 7.3.2** 令 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T} = \boldsymbol{D} + \boldsymbol{N}$ 是 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ 的 Schur 分解, 其中 $\boldsymbol{D}$ 是对角矩阵, $\boldsymbol{N}$ 是严格上三角形矩阵. 令 $\lambda$ 和 $\mu$ 分别代表 $\boldsymbol{A}$ 的绝对值最大和最小的特征值. 如果 $\theta \geqslant 0$, 则对于所有 $k \geqslant 0$ 我们有

$$
||\boldsymbol{A}^k||_2 \leqslant (1+\theta)^{n-1}\left(|\lambda| + \frac{||\boldsymbol{N}||_F}{1+\theta}\right)^k. \tag{7.3.11}
$$

如果 $\boldsymbol{A}$ 非奇异且 $\theta \geqslant 0$ 满足 $(1+\theta)|\mu| > ||\boldsymbol{N}||_F$, 则对所有 $k \geqslant 0$ 我们也有

$$
||\boldsymbol{A}^{-k}||_2 \leqslant (1+\theta)^{n-1}\left(\frac{1}{|\mu| - ||\boldsymbol{N}||_F/(1+\theta)}\right)^k. \tag{7.3.12}
$$

**证明** 对 $\theta \geqslant 0$, 定义对角矩阵 $\boldsymbol{\Delta}$ 为

$$
\boldsymbol{\Delta} = \mathrm{diag}(1, (1+\theta), (1+\theta)^2, \cdots, (1+\theta)^{n-1})
$$

并注意到 $\kappa_2(\boldsymbol{\Delta}) = (1+\theta)^{n-1}$. 由于 $N$ 是严格上三角形矩阵, 容易证明 $||\boldsymbol{\Delta}\boldsymbol{N}\boldsymbol{\Delta}^{-1}||_F \leqslant ||\boldsymbol{N}||_F/(1+\theta)$. 所以,

$$
\begin{aligned}
||\boldsymbol{A}^k||_2 =& ||\boldsymbol{T}^k||_2 = ||\boldsymbol{\Delta}^{-1}(\boldsymbol{D} + \boldsymbol{\Delta}\boldsymbol{N}\boldsymbol{\Delta}^{-1})^k\boldsymbol{\Delta}||_2 \\
\leqslant& \kappa_2(\boldsymbol{\Delta})(||\boldsymbol{D}||_2 + ||\boldsymbol{\Delta}\boldsymbol{N}\boldsymbol{\Delta}^{-1}||_2)^k \\
\leqslant& (1+\theta)^{n-1}\left(|\lambda| + \frac{||\boldsymbol{N}||_F}{1+\theta}\right)^k.
\end{aligned}
$$

另一方面, 如果 $\boldsymbol{A}$ 非奇异且 $(1+\theta)|\mu| > ||\boldsymbol{N}||_F$, 则从 $||\boldsymbol{\Delta}\boldsymbol{D}^{-1}\boldsymbol{N}\boldsymbol{\Delta}^{-1}||_2 < 1$ 和引理 2.3.3 我们有

$$
\begin{aligned}
||\boldsymbol{A}^{-k}||_2 =& ||\boldsymbol{T}^{-k}||_2 = ||\boldsymbol{\Delta}^{-1}[(\boldsymbol{I} + \boldsymbol{\Delta}\boldsymbol{D}^{-1}\boldsymbol{N}\boldsymbol{\Delta}^{-1})^{-1}\boldsymbol{D}^{-1}]^k\boldsymbol{\Delta}||_2 \\
\leqslant& \kappa_2(\boldsymbol{\Delta})\left(\frac{||\boldsymbol{D}^{-1}||_2}{1 - ||\boldsymbol{\Delta}\boldsymbol{D}^{-1}\boldsymbol{N}\boldsymbol{\Delta}^{-1}||_2}\right)^k \\
\leqslant& (1+\theta)^{n-1}\left(\frac{1}{|\mu| - ||\boldsymbol{N}||_F/(1+\theta)}\right)^k. \quad \square
\end{aligned}
$$

**定理 7.3.1 的证明**

用归纳法容易推知 $A^kQ_0 = Q_k(R_k\cdots R_1)$. 将 (7.3.5) 和 (7.3.6) 代入此等式有

$$T^k\begin{bmatrix} V_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} V_k \\ W_k \end{bmatrix}(R_k\cdots R_1),$$

其中 $V_k = Q_\alpha^{\mathrm{H}}Q_k$, $W_k = Q_\beta^{\mathrm{H}}Q_k$. 利用引理 7.1.5 得, 存在矩阵 $X\in\mathbb{C}^{r\times(n-r)}$ 使得

$$\begin{bmatrix} I_r & X \\ 0 & I_{n-r} \end{bmatrix}^{-1}\begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}\begin{bmatrix} I_r & X \\ 0 & I_{n-r} \end{bmatrix} = \begin{bmatrix} T_{11} & 0 \\ 0 & T_{22} \end{bmatrix},$$

于是

$$\begin{bmatrix} T_{11}^k & 0 \\ 0 & T_{22}^k \end{bmatrix}\begin{bmatrix} V_0 - XW_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} V_r - XW_k \\ W_k \end{bmatrix}(R_k\cdots R_1).$$

以下我们证明矩阵 $V_0 - XW_0$ 非奇异, 这使我们能获得以下表达式:

$$W_k = T_{22}^{\mathrm{k}}W_0(V_0 - XW_0)^{-1}T_{11}^{-k}[I_r, -X]\begin{bmatrix} V_k \\ W_k \end{bmatrix},$$

回忆 2.6.3 节中子空间之间距离的定义:

$$\mathrm{dist}(D_r(A), \mathrm{ran}(Q_k)) = \|Q_\beta^{\mathrm{H}}Q_k\|_2 = \|W_k\|_2.$$

由于 $\|[I_r, -X]\|_2 \leqslant 1 + \|X\|_F$, 我们有

$$\mathrm{dist}(D_r(A), \mathrm{ran}(Q_k)) \leqslant \|T_{22}^k\|_2\|(V_0 - XW_0)^{-1}\|_2\|T_{11}^{-k}\|_2(1 + \|X\|_F). \quad (7.3.13)$$

为证明定理, 我们必须对上界中的 4 项都要估计.

由于 $\mathrm{sep}(T_{11}, T_{22})$ 是线性变换 $\phi(X) = T_{11}X - XT_{22}$ 的最小奇异值, 容易从 $\phi(X) = -T_{12}$ 推得

$$\|X\|_F \leqslant \frac{\|T_{12}\|_F}{\mathrm{sep}(T_{11}, T_{22})}. \quad (7.3.14)$$

应用引理 7.3.2, 可证

$$\|T_{22}^k\|_2 \leqslant (1+\theta)^{n-r-1}\left(|\lambda_{r+1}| + \frac{\|N\|_F}{1+\theta}\right)^k, \quad (7.3.15)$$

$$\|T_{11}^{-k}\|_2 \leqslant (1+\theta)^{r-1}\Big/\left(|\lambda_r| - \frac{\|N\|_F}{1+\theta}\right)^k. \quad (7.3.16)$$

最后, 我们将考虑 $\|(V_0 - XW_0)^{-1}\|$. 注意到

$$V_0 - XW_0 = Q_\alpha^{\mathrm{H}}Q_0 - XQ_\beta^{\mathrm{H}}Q_0 = [I_r, -X]\begin{bmatrix} Q_\alpha^{\mathrm{H}} \\ Q_\beta^{\mathrm{H}} \end{bmatrix}Q_0$$

$$
\begin{aligned}
&= \left[ [\boldsymbol{Q}_\alpha \quad \boldsymbol{Q}_\beta] \begin{bmatrix} \boldsymbol{I}_r \\ -\boldsymbol{X}^{\mathrm{H}} \end{bmatrix} \right]^{\mathrm{H}} \boldsymbol{Q}_0 \\
&= (\boldsymbol{I}_r + \boldsymbol{X}\boldsymbol{X}^{\mathrm{H}})^{1/2} (\boldsymbol{Z}^H \boldsymbol{Q}_0),
\end{aligned}
$$

其中

$$
\begin{aligned}
\boldsymbol{Z} &= [\boldsymbol{Q}_\alpha \quad \boldsymbol{Q}_\beta] \begin{bmatrix} \boldsymbol{I}_r \\ -\boldsymbol{X}^{\mathrm{H}} \end{bmatrix} (\boldsymbol{I}_r + \boldsymbol{X}\boldsymbol{X}^{\mathrm{H}})^{-1/2} \\
&= (\boldsymbol{Q}_\alpha - \boldsymbol{Q}_\beta \boldsymbol{X}^{\mathrm{H}})(\boldsymbol{I}_r + \boldsymbol{X}\boldsymbol{X}^{\mathrm{H}})^{-1/2}.
\end{aligned}
$$

这个矩阵的列是正交的. 它们也是 $D_r(\boldsymbol{A}^{\mathrm{H}})$ 的基, 因为

$$
\boldsymbol{A}^{\mathrm{H}}(\boldsymbol{Q}_\alpha - \boldsymbol{Q}_\beta \boldsymbol{X}^{\mathrm{H}}) = (\boldsymbol{Q}_\alpha - \boldsymbol{Q}_\beta \boldsymbol{X}^{\mathrm{H}}) \boldsymbol{T}_{11}^{\mathrm{H}}.
$$

此关系式可从等式 $\boldsymbol{A}^{\mathrm{H}}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{T}^{\mathrm{H}}$ 推出.

从定理 2.6.1 知

$$
d = \mathrm{dist}(D_r(\boldsymbol{A}^{\mathrm{H}}), \mathrm{ran}(\boldsymbol{Q}_0)) = \sqrt{1 - \sigma_r(\boldsymbol{Z}^{\mathrm{H}}\boldsymbol{Q}_0)^2}
$$

且由假设 $d < 1$,

$$
\sigma_r(\boldsymbol{Z}^{\mathrm{H}}\boldsymbol{Q}_0) > 0.
$$

这表明

$$
(\boldsymbol{V}_0 - \boldsymbol{X}\boldsymbol{W}_0) = (\boldsymbol{I}_r + \boldsymbol{X}\boldsymbol{X}^{\mathrm{H}})^{1/2} (\boldsymbol{Z}^{\mathrm{H}}\boldsymbol{Q}_0)
$$

非奇异, 于是,

$$
\begin{aligned}
\|(\boldsymbol{V}_0 - \boldsymbol{X}\boldsymbol{W}_0)^{-1}\|_2 &\leqslant \|(\boldsymbol{I}_r + \boldsymbol{X}\boldsymbol{X}^{\mathrm{H}})^{-1/2}\|_2 \|(\boldsymbol{Z}^{\mathrm{H}}\boldsymbol{Q}_0)^{-1}\|_2 \\
&\leqslant 1/\sqrt{1 - d^2} \qquad (7.3.17)
\end{aligned}
$$

将 (7.3.14)~(7.3.17) 代入到 (7.3.13) 即得到定理. □

## 习　　题

**7.3.1** (a) 证明: 若 $\boldsymbol{X} \in \mathbb{C}^{n \times n}$ 非奇异, 则 $\|\boldsymbol{A}\|_X = \|\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}\|_2$ 定义了一个矩阵范数, 具有性质 $\|\boldsymbol{A}\boldsymbol{B}\|_X \leqslant \|\boldsymbol{A}\|_X \|\boldsymbol{B}\|_X$. (b) 令 $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ 且设 $\rho = \max|\lambda_i|$. 证明: 对任何 $\varepsilon > 0$, 存在非奇异矩阵 $\boldsymbol{X} \in \mathbb{C}^{n \times n}$ 使得 $\|\boldsymbol{A}\|_X = \|\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}\|_2 \leqslant \rho + \varepsilon$. 由此得出结论: 存在一个常数 $M$, 使得 $\|\boldsymbol{A}^k\|_2 \leqslant M(\rho + \varepsilon)^k$ 对所有非负整数 $k$ 均成立. (提示: 设 $\boldsymbol{X} = \boldsymbol{Q}\mathrm{diag}(1, a, \cdots, a^{n-1})$, 其中 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{D} + \boldsymbol{N}$ 是 $\boldsymbol{A}$ 的 Schur 分解.)

**7.3.2** 证明: 由 (7.3.10) 式计算得到的矩阵即为由 (7.3.9) 所定义的 $\boldsymbol{T}_k$.

**7.3.3** 设 $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ 非奇异且 $\boldsymbol{Q}_0 \in \mathbb{C}^{n \times p}$ 列正交. 下列迭代称为逆正交迭代:

**for** $k = 1, 2, \cdots$

　　从 $AZ_k = Q_{k-1}$解出$Z_k \in \mathbb{C}^{n \times p}$

　　$Z_k = Q_k R_k$　(QR分解)

**end**

试解释为什么此迭代通常用来计算 $A$ 的 $p$ 个绝对值最小的特征值. 注意到实现此迭代, 有必要解以 $A$ 为系数的线性方程组. 当 $p=1$ 时, 方法称为*逆幂法*.

**7.3.4** 设 $A \in \mathbb{R}^{n\times n}$ 有特征值 $\lambda_1, \cdots, \lambda_n$ 满足

$$\lambda = \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 > |\lambda_5| \geqslant \cdots \geqslant |\lambda_n|,$$

其中 $\lambda$ 为正数. 设 $A$ 有两个形如

$$\begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix}$$

的 Jordan 块. 试讨论当用幂法求此矩阵时, 幂法的收敛性. 讨论如何对此收敛进行加速.

## 本节注释与参考文献

关于幂法实用的详细讨论在 Wilkinson(1965, 第 10 章) 中给出. 方法讨论了怎样加速基本迭代, 计算非主特征值以及处理复共轭特征值. 下面文章讨论了不同幂迭代法之间的联系:

B. N. Parlett and W. G. Poole(1973). "A. Geometric Theory for the QR, LU, and Power Iterations," *SIAM J. Num. Anal,* 10, 389–412.

QR 迭代在下面两文章中同时提出来的.

J. G. F. Francis(1961). "The QR Transformation: A Unitary Analogue to the LR Transformation," *Com. J.* 4, 265–271, 332–334.

V. N. Kublanovskaya(1961). "On Some Algorithms for the Soultion of the Complete Eigenvalue Problem," *USSR Comp. Math. Phys.* 3, 637–657.

由第一篇的标题可知, LR 迭代比 QR 迭代更早发明. 前者的基本算法由下文提出:

H. Rutishauser(1958). "Solution of Eigenvalue Problems with the LR Transformation," *Nat. Bur. Stand. App. Math. Ser.* 49, 47–81.

B. N. Parlett(1995). "The New qd Algorithms," *ACTA Numerica* 5, 459–491

出现了很多关于 QR 迭代法的收敛性的文章. 其中几篇是:

J. H. Wilkinson(1965). "Convergence of the LR, QR, and Related Algorithms," *Comp. J.* 8, 77–84.

B. N. Parlett(1965). "Convergence of the QR Algorithm," *Numer. Math.* 7, 187–193. (Correction in *Numer. Math.* 10, 163–164.)

B. N. Parlett(1966). "Singular and Invariant Matrices Under the QR Algorithm," *Math. Comp.* 20, 611–615.

B. N. Parlett(1968). "Global Convergence of the Basic QR Algorithm on Hessenberg Matrices," *Math. Comp.* 22, 803–817.

Wilkinson(AEP, 第 9 章) 也讨论了这一重要算法的收敛性理论, 欲深入了解 QR 算法的收敛性及它与别的重要算法的联系, 可以读以下文章:

D. S. Watkins(1982). "Understanding the QR Algorithm," *SIAM Review* 24, 427–440.

T. Nanda(1985). "Differential Equations and the QR Algorithm," *SIAM J. Numer. Anal.* 22, 310–321.

D. S. Watkins(1993). "Some Perspectives on the Eigenvalue Problem," *SIAM Review* 35, 430–471.

下面文章有关同步迭代的一些实际和理论性质:

H. Rutishauser(1970). "Simultaneous Iteration Method for Symmetric Matrices," *Numer. Math.* 16, 205–223. See also(Wilkinson and Reinsch) 1971, pp. 284–302.

M. Clint and A. Jennings(1971). "A Simultaneous Iteration Method for the Unsymmetric Eigenvalue Problem," *J. Inst. Math. Applic.* 8, 111–121.

A. Jennings and D. R. L. Orr(1971). "Application of the Simultaneous Iteration Method to Undamped Vibration Problems," *Inst. J. Numer. Math. Eng.* 3, 13–24.

A. Jennings and W. J. Stewart(1975). "Simultaneous Iteration for the Partial Eigensolution of Real Matrices," *J. Inst. Math. Applic.* 15, 351–362.

G. W. Stewart(1975). "Methods of Simultaneous Iteration for Calculation Eigenvectors of Matrices," in *Topics in Numerical Analysis II*, ed. John J. H. Miller, Academic Press, New York, pp. 185–196.

G. W. Stewart(1976). "Simultaneous Iteration for Computing Invariant Subspaces of Non-Hermitian Matrices," *Mumer. Math.* 25, 123–136.

也可参阅以下专著的第 10 章:

A. Jennings(1977). *Matrix Computation for Engineers and Scientists,* John Wiley and Sons, New York.

同步迭代和 Lanczos 算法 (参阅第 9 章) 是求一般稀疏矩阵的几个特征值的主要方法.

## 7.4 Hessenberg 分解和实 Schur 型

在本节和下一节, 我们阐述怎样使 QR 迭代 (7.3.1) 成为计算 Schur 分解的快速、有效方法. 因为大部分特征值和不变子空间问题只涉及实数据, 所以我们重点推导 (7.3.1) 的实形式:

$$
\begin{aligned}
&H_0 = U_0^{\mathrm{T}} A U_0 \\
&\textbf{for}\quad k = 1, 2, \cdots \\
&\qquad H_{k-1} = U_k R_k \quad (\text{QR分解}) \\
&\qquad H_k = R_k U_k \\
&\textbf{end}
\end{aligned}
\tag{7.4.1}
$$

其中 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 每个 $\boldsymbol{U}_k \in \mathbb{R}^{n\times n}$ 是正交矩阵, 每个 $\boldsymbol{R}_k \in \mathbb{R}^{n\times n}$ 是上三角形矩阵. 与实迭代相关的困难是 $\boldsymbol{H}_k$ 不会收敛到严格的、"特征值暴露" 的三角形矩

阵, 因为 $\boldsymbol{A}$ 有复特征值. 由于这个原因, 我们必须降低期望值, 而满足于计算称为实Schur分解的另一种分解.

为了有效计算实 Schur 分解, 我们必须谨慎选取 (7.4.1) 中的初始正交相似变换矩阵 $\boldsymbol{U}_0$. 特别地, 如果我们选择 $\boldsymbol{U}_0$ 使得 $\boldsymbol{H}_0$ 是上 Hessenberg 矩阵, 那么每次迭代工作量将会从 $O(n^3)$ 减为 $O(n^2)$. 最初约化 Hessenberg 型 (计算 $\boldsymbol{U}_0$) 本身是很重要的计算, 它可通过一系列 Householder 矩阵运算来实现.

### 7.4.1　实 Schur 分解

对角块均为 1×1 或 2×2 的分块上三角形矩阵称为拟上三角形矩阵. 实 Schur 分解相当于将矩阵实约化为一个拟上三角型.

**定理 7.4.1(实 Schur 分解)**　*若 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 则存在一个正交矩阵 $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ 使得*

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} & \cdots & \boldsymbol{R}_{1m} \\ \boldsymbol{0} & \boldsymbol{R}_{22} & \cdots & \boldsymbol{R}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{R}_{mm} \end{bmatrix}, \tag{7.4.2}$$

*其中每个 $\boldsymbol{R}_{ii}$ 不是 1×1 矩阵就是有复共轭特征值的 2×2 矩阵.*

**证明**　$\boldsymbol{A}$ 的复特征值一定以共轭对出现, 因为特征多项式 $\det(z\boldsymbol{I}-\boldsymbol{A})$ 的系数都是实数. 令 $k$ 是 $\lambda(\boldsymbol{A})$ 中复共轭对个数. 我们对 $k$ 采用归纳法证明定理. 首先注意到引理 7.1.2 和定理 7.1.3 有实的类似结论. 这样, 当 $k=0$ 时, 定理成立. 现在假设 $k \geqslant 1$. 如果 $\lambda = \gamma + \mathrm{i}\mu \in \lambda(\boldsymbol{A})$ 且 $\mu \neq 0$, 那么在 $\mathbb{R}^n$ 中存在向量 $\boldsymbol{y}$ 和 $\boldsymbol{z}$ ($z \neq \boldsymbol{0}$) 使得 $\boldsymbol{A}(y+\mathrm{i}z) = (\gamma+\mathrm{i}\mu)(y+\mathrm{i}z)$, 即

$$\boldsymbol{A}[\boldsymbol{y}\ \ \boldsymbol{z}] = [\boldsymbol{y}\ \ \boldsymbol{z}]\begin{bmatrix} \gamma & \mu \\ -\mu & \gamma \end{bmatrix}.$$

假设 $\mu \neq 0$ 表明 $\boldsymbol{y}$ 与 $\boldsymbol{z}$ 张成 $\boldsymbol{A}$ 的一个二维实不变子空间. 从引理 7.1.2 可推得, 存在正交矩阵 $\boldsymbol{U} \in \mathbb{R}^{n\times n}$, 使得

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_{22} \end{bmatrix} \begin{matrix} 2 \\ n-2 \end{matrix}$$
$$\qquad\qquad\quad 2 \quad\ n-2$$

其中 $\lambda(\boldsymbol{T}_{11}) = \{\lambda, \bar{\lambda}\}$. 由归纳法, 存在一个正交矩阵 $\tilde{\boldsymbol{U}}$ 使得 $\tilde{\boldsymbol{U}}^{\mathrm{T}}\boldsymbol{T}_{22}\tilde{\boldsymbol{U}}$ 有所需之结构. 令 $\boldsymbol{Q} = \boldsymbol{U}\mathrm{diag}(\boldsymbol{I}_2, \tilde{\boldsymbol{U}})$ 即知定理成立. □

定理表明任一实矩阵可以正交相似于一个拟上三角形矩阵. 很明显, 复特征值的实部和虚部容易由 2 × 2 的对角块获得.

### 7.4.2　Hessenberg QR 步

现在, 我们把注意力转移到怎样加速计算 (7.4.1) 中单个 QR 步. 关于此, (7.4.1) 的最大缺点是每步都需进行花费 $O(n^3)$ 个 flop 的完全 QR 分解. 幸运的是, 若

明智地选取正交矩阵 $U_0$, 每次迭代的工作量可降低一个数量级. 特别地, 如果 $\boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U}_0 = \boldsymbol{H}_0 = (h_{ij})$ 是上 Hessenberg 矩阵 ($h_{ij} = 0, i > j+1$), 那么以后计算每个 $\boldsymbol{H}_k$ 只需 $O(n^2)$ 个 flop. 为说明这点, 当 $\boldsymbol{H}$ 是上 Hessenberg 矩阵时, 我们来看看计算 $\boldsymbol{H} = \boldsymbol{QR}$ 和 $\boldsymbol{H}_+ = \boldsymbol{RQ}$. 如 5.2.4 节所述, 我们通过 $n-1$ 个 Givens 旋转变换: $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{H} \equiv \boldsymbol{G}_{n-1}^{\mathrm{T}}\cdots\boldsymbol{G}_1^{\mathrm{T}}\boldsymbol{H} = \boldsymbol{R}$ 可以将 $\boldsymbol{H}$ 化为上三角形矩阵. 这里 $\boldsymbol{G}_i = G(i, i+1, \theta_i)$. 对于 $n=4$ 情形, 作三次 Givens 左乘:

$$
\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}
$$

$$
\longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}.
$$

参见算法 5.2.3.

计算 $\boldsymbol{RQ} = \boldsymbol{R}(\boldsymbol{G}_1\cdots\boldsymbol{G}_{n-1})$ 同样容易实现. 在 $n=4$ 时作三次 Givens 右乘:

$$
\begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}
$$

$$
\longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \longrightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}.
$$

综上所述, 我们得到下面的算法.

**算法 7.4.1** 如果 $\boldsymbol{H}$ 是 $n \times n$ 上 Hessenberg 矩阵, 则本算法用 $\boldsymbol{H}_+ = \boldsymbol{RQ}$ 覆盖 $\boldsymbol{H}$, 这里 $\boldsymbol{H} = \boldsymbol{QR}$ 是 $\boldsymbol{H}$ 的 QR 分解.

**for** $k = 1 : n-1$

$\qquad [c(k), s(k)] = \textbf{givens}(H(k,k), H(k+1,k))$

$$H(k:k+1, k:n) = \begin{bmatrix} c(k) & s(k) \\ -s(k) & c(k) \end{bmatrix}^{\mathrm{T}} H(k:k+1, k:n)$$

**end**

**for** $k = 1 : n-1$

$$H(1:k+1, k:k+1) = H(1:k+1, k:k+1)\begin{bmatrix} c(k) & s(k) \\ -s(k) & c(k) \end{bmatrix}$$

**end**

令 $\boldsymbol{G}_k = G(k, k+1, \theta_k)$ 是第 $k$ 次 Givens 旋转变换. 容易断定 $\boldsymbol{Q} = \boldsymbol{G}_1 \cdots \boldsymbol{G}_{n-1}$ 是上 Hessenberg 矩阵. 这样, $\boldsymbol{RQ} = \boldsymbol{H}_+$ 也是上 Hessenberg 矩阵. 这个算法需大约 $6n^2$ 个 flop, 这就比 (7.3.1) 中的满矩阵 QR 步快一个数量级.

**例 7.4.1** 将算法 7.4.1 应用于

$$\boldsymbol{H} = \begin{bmatrix} 3 & 1 & 2 \\ 4 & 2 & 3 \\ 0 & 0.01 & 1 \end{bmatrix},$$

则有

$$\boldsymbol{G}_1 = \begin{bmatrix} 0.6 & -0.8 & 0 \\ 0.8 & 0.6 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\boldsymbol{G}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.9996 & -0.0249 \\ 0 & 0.0249 & 0.9996 \end{bmatrix},$$

$$\boldsymbol{H}_+ = \begin{bmatrix} 4.7600 & -2.5442 & 5.4653 \\ 0.3200 & 0.1856 & -2.1796 \\ 0.0000 & 0.0263 & 1.0540 \end{bmatrix}.$$

### 7.4.3 Hessenberg 约化

接下来要说明的是如何计算 Hessenberg分解:

$$\boldsymbol{U}_0^{\mathrm{T}} \boldsymbol{A} \boldsymbol{U}_0 = \boldsymbol{H}, \quad \boldsymbol{U}_0^{\mathrm{T}} \boldsymbol{U}_0 = \boldsymbol{I}. \tag{7.4.3}$$

变换矩阵 $\boldsymbol{U}_0$ 能够通过计算 Householder 矩阵 $\boldsymbol{P}_1, \cdots \boldsymbol{P}_{n-2}$ 之乘积而得到. 矩阵 $P_k$ 的作用是将第 $k$ 列在次对角元以下的元素都化为零. 在 $n = 6$ 时, 我们有

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{P}_1} \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{P}_2}$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{P}_3} \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{P}_4}$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

通常, $k-1$ 步后, 我们计算了 $k-1$ 个 Householder 矩阵 $\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{k-1}$, 使得

$$(\boldsymbol{P}_1\cdots\boldsymbol{P}_{k-1})^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{P}_1\cdots\boldsymbol{P}_{k-1}) = \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{B}_{13} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23} \\ \boldsymbol{0} & \boldsymbol{B}_{32} & \boldsymbol{B}_{33} \end{bmatrix} \begin{matrix} k-1 \\ 1 \\ n-k \end{matrix}$$
$$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad k-1 \quad 1 \quad n-k$$

的前 $k-1$ 列是上 Hessenberg 矩阵. 假设 $\bar{\boldsymbol{P}}_k$ 是秩为 $n-k$ 的 Householder 矩阵, 使得 $\bar{\boldsymbol{P}}_k\boldsymbol{B}_{32}$ 是 $e_1^{(n-k)}$ 的倍数, 如果 $\boldsymbol{P}_k = \mathrm{diag}(\boldsymbol{I}_k, \bar{\boldsymbol{P}}_k)$, 那么

$$(\boldsymbol{P}_1\cdots\boldsymbol{P}_k)^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{P}_1\cdots\boldsymbol{P}_k) = \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{B}_{13}\bar{\boldsymbol{P}}_k \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23}\bar{\boldsymbol{P}}_k \\ \boldsymbol{0} & \bar{\boldsymbol{P}}_k\boldsymbol{B}_{32} & \bar{\boldsymbol{P}}_k\boldsymbol{B}_{33}\bar{\boldsymbol{P}}_k \end{bmatrix}$$

的前 $k$ 列为上 Hessenberg 矩阵. 对于 $k=1:n-2$ 重复此一过程, 我们得到下列算法.

**算法 7.4.2(Householder 约化矩阵为 Hessenberg 型)** 给定 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 下面的算法计算 $\boldsymbol{H} = \boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U}_0$ 并覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{H}$ 是上 Hessenberg 矩阵且 $\boldsymbol{U}_0$ 是 Householder 矩阵的乘积.

```
for k = 1 : n − 2
    [v, β] = house(A(k + 1 : n, k))
    A(k + 1 : n, k : n) = (I − βvv^T)A(k + 1 : n, k : n)
    A(1 : n, k + 1 : n) = A(1 : n, k + 1 : n)(I − βvv^T)
end
```

这一算法需 $10n^3/3$ 个 folp. 若要显式写出 $\boldsymbol{U}_0$, 需附加 $4n^3/3$ flop. 第 $k$ 个 Householder 矩阵可存放在 $A(k+2:n,k)$ 中. 欲知详情, 请见 Martin and Wilkinson(1968$d$).

这个方法化 $\boldsymbol{A}$ 为 Hessenberg 型的舍入误差性质是很令人满意的. Wilkinson(1965, 351 页) 指出计算的 Hessenberg 矩阵 $\hat{\boldsymbol{H}}$ 满足 $\hat{\boldsymbol{H}} = \boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{Q}$, 这里 $\boldsymbol{Q}$ 正交且 $\|\boldsymbol{E}\|_F \leqslant cn^2\mathbf{u}\|\boldsymbol{A}\|_F$, $c$ 为较小的常数.

**例 7.4.2** 若

$$\boldsymbol{A} = \begin{bmatrix} 1 & 5 & 7 \\ 3 & 0 & 6 \\ 4 & 3 & 1 \end{bmatrix}, \quad \boldsymbol{U}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.6 & 0.8 \\ 0 & 0.8 & -0.6 \end{bmatrix},$$

则

$$U_0^{\mathrm{T}} A U_0 = H = \begin{bmatrix} 1.00 & 8.60 & -0.20 \\ 5.00 & 4.96 & -0.27 \\ 0.00 & 2.28 & -3.96 \end{bmatrix}$$

### 7.4.4 三级性质

Hessenberg 约化 (算法 7.4.2) 有大量的 2 级运算：一半是 gaxpys 修正，一半是外积修正. 为了在约化过程中引入 3 级计算, 我们简略讨论两种方法.

第一种方法非常直接, 它涉及把分块矩阵约化为分块 Hessenberg 型. 设 (为清楚起见)$n = rN$ 并记

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$$
$$\qquad\quad r \qquad n-r$$

假设我们已计算 $A_{21}$ 的 QR 分解 $A_{21} = \bar{Q}_1 R_1$, 其中 $\bar{Q}_1$ 是 WY 型. 也就是说, 我们有 $W_1, Y_1 \in \mathbb{R}^{(n-r)\times r}$ 使得 $\bar{Q}_1 = I - W_1 Y_1^{\mathrm{T}}$ (参看 5.2.2 节). 若 $Q_1 = \mathrm{diag}(I_r, \bar{Q}_1)$, 则

$$Q_1^{\mathrm{T}} A Q_1 = \begin{bmatrix} A_{11} & A_{12}\bar{Q}_1 \\ R_1 & \bar{Q}_1^{\mathrm{T}} A_{22} \bar{Q}_1 \end{bmatrix}.$$

注意到, 只要 $\bar{Q}_1$ 是 WY 型, 那么修正 (1,2) 块和 (2,2) 块就有大量的 3 级运算. $Q_1^{\mathrm{T}} A Q_1$ 的第一块列是分块上 Hessenberg 型. 这也就完全演示了全过程. 接着我们重复计算 $\bar{Q}_1^{\mathrm{T}} A_{22} \bar{Q}_1$ 的前 $r$ 列. 在 $N-2$ 步之后, 我们获得

$$H = U_0^{\mathrm{T}} A U_0 = \begin{bmatrix} H_{11} & H_{12} & \cdots & \cdots & H_{1N} \\ H_{21} & H_{22} & \cdots & \cdots & H_{2N} \\ 0 & \ddots & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & H_{N,N-1} & H_{NN} \end{bmatrix}$$

其中每个 $H_{ij}$ 是 $r \times r$ 矩阵且 $U_0 = Q_1 \cdots Q_{N-2}$, 每个 $Q_i$ 都是 WY 型. 整个算法中 3 级计算所占比例为 $1 - O(1/N)$.

注意到 $H$ 中的次对角块是上三角形矩阵, 于是矩阵下带宽为 $p$. 除了第一条次对角线外, 其余都可以用 Givens 旋转变换消为零, 从而约化 $H$ 为一个真正的 Hessenberg 型.

Dongarra, Hammarling and Sorensen(1987) 已证明, 怎样混合使用 gaxpy 算法和 3 级修正直接约化为 Hessenberg 型. 他们的思想是当每个 Householder 变换已产生后, 只需要作极小化校正. 例如, 设第一个 Householder 矩阵 $P_1$ 已算出, 为得到 $P_2$, 我们只需要 $P_1 A P_1$ 的第 2 列, 不是整个外积修正. 为构造 $P_3$, 我们只需 $P_2 P_1 A P_1 P_2$ 的第 3 列, 等等. 用这种方式, 仅需用 gaxpy 运算就可确定

Householder 矩阵, 而不涉及任何外积修正. 一旦有了适量的 Householder 矩阵, 它们合在一起用 3 级方式去进行修正.

### 7.4.5 Hessenberg 矩阵的重要性质

Hessenberg 分解不是唯一的. 如果 $\boldsymbol{Z}$ 是任一 $n\times n$ 正交矩阵, 我们应用算法 7.4.2 于 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Z}$, 则 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 是上 Hessenberg 矩阵, 其中 $\boldsymbol{Q}=\boldsymbol{Z}\boldsymbol{U}_0$. 然而, $\boldsymbol{Q}\boldsymbol{e}_1=\boldsymbol{Z}(\boldsymbol{U}_0\boldsymbol{e}_1)=\boldsymbol{Z}\boldsymbol{e}_1$, 这表明只要 $\boldsymbol{Q}$ 的第 1 列是指定的, 则 $\boldsymbol{H}$ 就会唯一. 实质上这在 $\boldsymbol{H}$ 没有零次对角元素时是千真万确的. 具有这种性质的 Hessenberg 矩阵称为*不可约的*. 以下是一条很重要的定理, 它说明了 Hessenberg 约化的唯一性.

**定理 7.4.2 (隐式 Q 定理)** 假设 $\boldsymbol{Q}=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_n]$ 和 $\boldsymbol{V}=[\boldsymbol{v}_1,\cdots,\boldsymbol{v}_n]$ 都是正交矩阵, 且 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 和 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}=\boldsymbol{G}$ 均是上 Hessenberg 矩阵, 这里 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$. 令 $k$ 是使 $h_{k+1,k}=0$ 的最小正整数, 当 $\boldsymbol{H}$ 不可约时约定 $k=n$. 如果 $\boldsymbol{q}_1=\boldsymbol{v}_1$, 那么 $\boldsymbol{q}_i=\pm\boldsymbol{v}_i$ 且 $|h_{i,i-1}|=|g_{i,i-1}|$, $i=2:k$. 而且, 若 $k<n$, 则 $g_{k+1,k}=0$.

**证明** 定义正交矩阵 $\boldsymbol{W}=[\boldsymbol{w}_1,\cdots,\boldsymbol{w}_n]=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{Q}$ 且注意到 $\boldsymbol{G}\boldsymbol{W}=\boldsymbol{W}\boldsymbol{H}$. 通过比较这一等式的 $i-1$ 列 $(i=2:k)$, 我们知道

$$h_{i,i-1}\boldsymbol{w}_i=\boldsymbol{G}\boldsymbol{w}_{i-1}-\sum_{j=1}^{i-1}h_{j,i-1}\boldsymbol{w}_j.$$

由$\boldsymbol{w}_1=\boldsymbol{e}_1$, 得出$[\boldsymbol{w}_1,\cdots,\boldsymbol{w}_k]$是上三角形矩阵, 于是 $\boldsymbol{w}_i=\pm\boldsymbol{I}_n(:,i)=\pm\boldsymbol{e}_i, i=2:k$. 从 $\boldsymbol{w}_i=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{q}_i$ 和 $h_{i,i-1}=\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{G}\boldsymbol{w}_{i-1}$ 可以推出 $\boldsymbol{v}_i=\pm\boldsymbol{q}_i$ 且

$$|h_{i,i-1}|=|\boldsymbol{q}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_{i-1}|=|\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}_{i-1}|=|g_{i,i-1}|,\quad i=2:k.$$

如果 $k<n$, 则

$$\begin{aligned} g_{k+1,k} &= \boldsymbol{e}_{k+1}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{e}_k=\boldsymbol{e}_{k+1}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{W}\boldsymbol{e}_k=\boldsymbol{e}_{k+1}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{H}\boldsymbol{e}_k \\ &= \boldsymbol{e}_{k+1}^{\mathrm{T}}\sum_{i=1}^{k}h_{ik}\boldsymbol{W}\boldsymbol{e}_i=\sum_{i=1}^{k}h_{ik}\boldsymbol{e}_{k+1}^{\mathrm{T}}\boldsymbol{e}_i=0. \end{aligned}$$

□

隐式 Q 定理的要旨在于, 如果 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 和 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Z}=\boldsymbol{G}$ 都是不可约的上 Hessenberg 矩阵, 且 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 第一列相同, 那么 $\boldsymbol{G}$ 和 $\boldsymbol{H}$ 是 "本质上相等", 即 $\boldsymbol{G}=\boldsymbol{D}^{-1}\boldsymbol{H}\boldsymbol{D}$, 其中 $\boldsymbol{D}=\mathrm{diag}(\pm1,\cdots,\pm1)$.

下一个定理涉及称为 Krylov 矩阵的一种新类型矩阵. 如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 和 $\boldsymbol{v}\in\mathbb{R}^n$, 那么 Krylov 矩阵 $\boldsymbol{K}(\boldsymbol{A},\boldsymbol{v},j)\in\mathbb{R}^{n\times j}$ 定义为

$$\boldsymbol{K}(\boldsymbol{A},\boldsymbol{v},j)=[\boldsymbol{v}\quad \boldsymbol{A}\boldsymbol{v}\ \cdots\ \boldsymbol{A}^{j-1}\boldsymbol{v}].$$

可以证明在 Hessenberg 约化 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 和 Krylov 矩阵 $\boldsymbol{K}(\boldsymbol{A},\boldsymbol{Q}(:,1),n)$ 的 QR 分解之间有联系.

**定理 7.4.3** 假设 $\boldsymbol{Q}\in\mathbb{R}^{n\times n}$ 是正交矩阵且 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$. 则 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 是不可约上 Hessenberg 矩阵当且仅当 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{K}(\boldsymbol{A},\boldsymbol{Q}(:,1),n)=\boldsymbol{R}$ 非奇异且为上三角形矩阵.

**证明** 设 $Q \in \mathbb{R}^{n\times n}$ 为正交矩阵且令 $H = Q^{\mathrm{T}}AQ$. 考虑以下恒等式

$$Q^{\mathrm{T}}K(A, Q(:,1), n) = [e_1 \quad He_1 \ \cdots \ H^{n-1}e_1] \equiv R.$$

如果 $H$ 是不可约的上 Hessenberg 矩阵, 那么很清楚 $R$ 是上三角形矩阵且 $r_{ii} = h_{21}h_{32}\cdots h_{i,i-1}, i = 2:n$. 由 $r_{11} = 1$ 可知 $R$ 非奇异. □

为了证明其逆关系也成立, 设 $R$ 是上三角形矩阵且非奇异. 由于 $R(:,k+1) = HR(:,k)$, 可以得出 $H(:k) \in \mathrm{span}\{e_1, \cdots, e_{k+1}\}$. 这隐含着 $H$ 是上 Hessenberg 矩阵. 由于 $r_{nn} = h_{21}h_{32}\cdots h_{n,n-1} \neq 0$, 可以推出 $H$ 是不可约的.

这样, 在非奇异 Krylov 矩阵和将矩阵约化为不可约的 Hessenberg 矩阵的正交相似变换之间多多少少有一种对应关系. 最后一个结论是有关不可约上 Hessenberg 矩阵的特征值问题的.

**定理 7.4.4** *如果 $\lambda$ 是不可约的上 Hessenberg 矩阵 $H \in \mathbb{R}^{n\times n}$ 的特征值, 那么它的几何重数为 1.*

**证明** 由于 $H - \lambda I$ 的前 $n-1$ 列线性无关, 因而对任意 $\lambda \in \mathbb{C}$, 我们有 $\mathrm{rank}(A - \lambda I) \geqslant n-1$. □

### 7.4.6 友矩阵型

正如 Schur 分解有相应的非酉 Jordan 分解一样, Hessenberg 分解也有其相应的非酉*友矩阵分解*. 令 $x \in \mathbb{R}^n$ 且设 Krylov 矩阵 $K = K(A, x, n)$ 非奇异. 如果 $c = c(0:n-1)$ 是线性方程组 $Kc = -A^n x$ 之解, 则可以推出 $AK = KC$, 其中

$$C = \begin{bmatrix} 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & \cdots & 0 & -c_2 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -c_{n-1} \end{bmatrix}. \tag{7.4.4}$$

矩阵 $C$ 称为友矩阵. 因为

$$\det(zI - C) = c_0 + c_1 z + \cdots + c_{n-1}z^{n-1} + z^n,$$

所以可以推出, 若 $K$ 非奇异, 那么分解 $K^{-1}AK = C$ 显示 $A$ 的特征多项式. 这一性质以及 $C$ 的稀疏性导致了各个应用领域的 "友矩阵方法". 这些方法一般包括:

- *计算 Hessenberg 分解 $U_0^{\mathrm{T}}AU_0 = H$;*
- *希望 $H$ 不可约并令 $Y = [e_1, He_1, \cdots, H^{n-1}e_1]$.*
- *由方程 $YC = HY$ 解出 $C$.*

不幸的是, 这个计算非常不稳定. 仅当 $A$ 的每个特征值的几何重数都为 1 时, $A$ 才相似于不可约 Hessenberg 矩阵. 具有这种性质的矩阵称为*非减阶矩阵*. 可以得知, 当 $A$ 靠近减阶矩阵时, 上述矩阵 $Y$ 的条件可能很差.

关于友矩阵计算的危险性的完整讨论, 可见 Wilkinson(1965, 从 405 页起)

### 7.4.7 用高斯变换进行 Hessenberg 约化

虽然是在讨论用非正交变换将矩阵约化为 Hessenberg 型, 我们应该指出也可用高斯变换取代算法 7.4.2 中的 Householder 矩阵. 特别地, 假设置换 $\boldsymbol{\Pi}_1, \cdots, \boldsymbol{\Pi}_{k-1}$ 和高斯变换 $\boldsymbol{M}_1, \cdots \boldsymbol{M}_{k-1}$ 已经确定, 它们使得

$$(\boldsymbol{M}_{k-1}\boldsymbol{\Pi}_{k-1}\cdots\boldsymbol{M}_1\boldsymbol{\Pi}_1)\boldsymbol{A}(\boldsymbol{M}_{k-1}\boldsymbol{\Pi}_{k-1}\cdots\boldsymbol{M}_1\boldsymbol{\Pi}_1)^{-1} = \boldsymbol{B},$$

其中

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{B}_{13} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23} \\ \boldsymbol{0} & \boldsymbol{B}_{32} & \boldsymbol{B}_{33} \end{bmatrix} \begin{matrix} k-1 \\ 1 \\ n-k \end{matrix}$$
$$\qquad\qquad k-1 \quad 1 \quad n-k$$

的前 $k-1$ 列是上 Hessenberg 矩阵. 然后确定一个秩为 $n-k$ 的置换 $\bar{\boldsymbol{\Pi}}_k$ 使得 $\bar{\boldsymbol{\Pi}}_k\boldsymbol{B}_{32}$ 的第一个元素绝对值最大. 这就有可能确定一个稳定的秩也为 $n-k$ 的高斯变换 $\bar{M}_k = \boldsymbol{I} - z_k e_1^{\mathrm{T}}$, 使得 $\bar{\boldsymbol{M}}_k(\bar{\boldsymbol{\Pi}}\boldsymbol{B}_{32})$ 的除第一个元素外的所有元素均为零. 定义 $\boldsymbol{\Pi}_k = \mathrm{diag}(\boldsymbol{I}_k, \bar{\boldsymbol{\Pi}}_k)$ 和 $\boldsymbol{M}_k = \mathrm{diag}(\boldsymbol{I}_k, \bar{\boldsymbol{M}}_k)$, 我们看到

$$(\boldsymbol{M}_k\boldsymbol{\Pi}_k\cdots\boldsymbol{M}_1\boldsymbol{\Pi}_1)\boldsymbol{A}(\boldsymbol{M}_k\boldsymbol{\Pi}_k\cdots\boldsymbol{M}_1\boldsymbol{\Pi}_1)^{-1}$$
$$= \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{B}_{13}\bar{\boldsymbol{\Pi}}_k^{\mathrm{T}}\bar{\boldsymbol{M}}_k^{-1} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23}\bar{\boldsymbol{\Pi}}_k^{\mathrm{T}}\bar{\boldsymbol{M}}_k^{-1} \\ \boldsymbol{0} & \bar{\boldsymbol{M}}_k\bar{\boldsymbol{\Pi}}_k\boldsymbol{B}_{32} & \bar{\boldsymbol{M}}_k\bar{\boldsymbol{\Pi}}_k\boldsymbol{B}_{33}\bar{\boldsymbol{\Pi}}_k^{\mathrm{T}}\bar{\boldsymbol{M}}_k^{-1} \end{bmatrix}$$

的前 $k$ 列是上 Hessenberg 矩阵. 注意到 $\bar{\boldsymbol{M}}_k^{-1} = \boldsymbol{I} + z_k e_1^{\mathrm{T}}$, 即知约化过程只是一些很简单的秩 1 校正.

仔细统计即知, 用高斯变换化为 Hessenberg 型只需 Householder 方法一半的运算量. 然而, 与列选主高斯消去法一样, 数据增长可能达到 $2^n$ 倍 (虽然少见). 见 Businger(1969). 用高斯方法的另一困难是, 特征值的条件数 (即 $s(\lambda)^{-1}$) 在非正交相似变换中并非保持不变, 这就使得误差估计变得复杂.

## 习　　题

**7.4.1** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 $\boldsymbol{z} \in \mathbb{R}^n$, 请给出一个详细算法来计算正交矩阵 $\boldsymbol{Q}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$ 是上 Hessenberg 矩阵且 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{z}$ 是 $\boldsymbol{e}_1$ 的倍量. (提示: 先约化 $z$, 然后应用算法 7.4.2)

**7.4.2** 详述用高斯变换化矩阵为 Hessenberg 型的整个过程并证明它只需 $5n^3/3$ 个 flop.

**7.4.3** 在一些情形, 有必要针对许多不同的 $z \in \mathbb{R}$ 和 $\boldsymbol{b} \in \mathbb{R}^n$ 的值求解线性方程组 $(\boldsymbol{A} + z\boldsymbol{I})\boldsymbol{x} = \boldsymbol{b}$. 说明用 Hessenberg 分解能有效且稳定地解决此问题.

**7.4.4** 给一个显式计算算法 7.4.2 中矩阵 $\boldsymbol{U}_0$ 的详细算法, 算法中 $\boldsymbol{H}$ 最终由 $\boldsymbol{U}_0$ 覆盖.

**7.4.5** 设 $\boldsymbol{H} \in \mathbb{R}^{n\times n}$ 是不可约上 Hessenberg 矩阵, 证明存在一对角矩阵 $\boldsymbol{D}$ 使得 $\boldsymbol{D}^{-1}\boldsymbol{H}\boldsymbol{D}$ 的每个次对角元都等于 1. $\kappa_2(\boldsymbol{D})$ 是多少?

**7.4.6** 设 $\boldsymbol{W},\boldsymbol{Y}\in\mathbb{R}^{n\times n}$, 且定义矩阵 $\boldsymbol{C}$ 和 $\boldsymbol{B}$ 为

$$\boldsymbol{C}=\boldsymbol{W}+\mathrm{i}\boldsymbol{Y},\quad \boldsymbol{B}=\begin{bmatrix}\boldsymbol{W} & -\boldsymbol{Y}\\ \boldsymbol{Y} & \boldsymbol{W}\end{bmatrix}.$$

证明如果 $\lambda\in\lambda(c)$ 为实, 则 $\lambda\in\lambda(\boldsymbol{B})$, 并说明相应特征向量的关系.

**7.4.7** 设 $\boldsymbol{A}=\begin{bmatrix} w & x\\ y & z\end{bmatrix}$ 是实矩阵, 有特征值 $\lambda\pm\mathrm{i}\mu$, 其中 $\mu$ 非零, 给出一个算法稳定求出 $c=\cos(\theta)$ 和 $s=\sin(\theta)$ 使得

$$\begin{bmatrix} c & s\\ -s & c\end{bmatrix}^{\mathrm{T}}\begin{bmatrix} w & x\\ y & z\end{bmatrix}\begin{bmatrix} c & s\\ -s & c\end{bmatrix}=\begin{bmatrix}\lambda & \beta\\ \alpha & \lambda\end{bmatrix},$$

其中 $\alpha\beta=-\mu^2$.

**7.4.8** 设 $(\lambda,x)$ 为上 Hessenberg 矩阵 $\boldsymbol{H}\in\mathbb{R}^{n\times n}$ 的一已知特征值-特征向量对. 给出一算法计算正交矩阵 $\boldsymbol{P}$, 使得

$$\boldsymbol{P}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{P}=\begin{bmatrix}\lambda & \boldsymbol{w}^{\mathrm{T}}\\ \boldsymbol{0} & \boldsymbol{H}_1\end{bmatrix},$$

其中 $\boldsymbol{H}_1\in\mathbb{R}^{(n-1)\times(n-1)}$ 是上 Hessenberg 矩阵. $\boldsymbol{P}$ 应为 Givens 变换之乘积.

**7.4.9** 设 $\boldsymbol{H}\in\mathbb{R}^{n\times n}$ 下带宽为 $p$, 说明怎样计算 $\boldsymbol{Q}\in\mathbb{R}^{n\times n}$(Givens 旋转之乘积), 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Q}$ 是上 Hessenberg 矩阵. 需多少个 flop?

**7.4.10** 证明: 若$C$是具有不同特征值$\lambda_1,\cdots,\lambda_n$的友矩阵, 则 $\boldsymbol{V}\boldsymbol{C}\boldsymbol{V}^{-1}=\mathrm{diag}(\lambda_1,\cdots,\lambda_n)$, 其中

$$\boldsymbol{V}=\begin{bmatrix}1 & \lambda_1 & \cdots & \lambda_1^{n-1}\\ 1 & \lambda_2 & \cdots & \lambda_2^{n-1}\\ \vdots & \vdots & \ddots & \vdots\\ 1 & \lambda_n & \cdots & \lambda_n^{n-1}\end{bmatrix}.$$

## 本节注释与参考文献

实 Schur 分解最初出现在:

F. D. Murnaghan and A. Wintner(1931). "A Canonical Form for Real Matrices Under Orthogonal Transformations," *Proc. Nat. Acad. Sci.* 17, 417–420.

Wilkinson(1965, 第 6 章) 中给出了约化为 Hessenberg 型的详细方法. Householder 方法和高斯方法的 Algol 程序可见:

R. S. Martin and J. H. Wilkinson(1968). "Similarity Reduction of a General Matrix to Hessenberg Form," *Numer. Math.* 12, 349–368. See also Wilkinson and Reinsch(1971, pp. 339–358).

上述参考文献的 Algol 程序之 Fortan 版本在 Eispak 程序包中. Givens 变换也被用于计算 Hessenberg 分解, 见:

W. Rath(1982). "Fast Givens Rotations for Orthogonal Similarity," *Numer. Math.* 40, 47–56.

Hessenberg 约化的高性能计算在下列文献中讨论:

J. J. Dongarra, L.Kaufman, and S. Hammarling(1986). "Squeezing the Most Out of Eigenvalue Solvers on High Performance Computers, " *Lin. Alg. and Its Applic.* 77, 113–136.

J. J. Dongarra, S. Hammarling, and D. C. Sorenen(1989). "Block Reduction of Matrices to Condensed Forms for Eigenvalue Computations, " *JACM* 27, 215–227.

M. W. Berry, J. J. Dongarra, and Y. Kim(1995). "A Parallel Algorithm for the Reduction of a Nonsymmetric Matrix to Block Upper Hessenberg Form, " *Parallel Computing* 21, 1189–1211.

高斯变换方法中指数增长的可能性在下列文献中首次出现:

P. Businger(1969). "Reducting a Matrix to Hessenberg Form, " *Math. Comp.* 23, 819–821.

然而, 我们应像对列选主元高斯消去法一样来看待此算法, 它在实际中是稳定的, 见 Eispack, 56–58 页.

有关稀疏矩阵的 Hessenberg 分解之讨论可见:

I. S. Duff and J. K. Reid(1975). "On the Reduction of Sparse Matrices to Condensed Forms by Similarity Transformations," *J. Inst. Math. Applic.* 15, 217–224.

一旦知道不可约上 Hessenberg 矩阵的特征值, 就可用 Givens 变换将次对角线上最后一个元素化为零. 见:

P. A. Businger(1971). "Numerically Stable Deflation of Hessenberg and Symmetric Tridiagonal Matrices, *BIT* 11, 262–270.

有关 Hessenberg 型的一些有趣数学性质, 可在下文中找到:

B. N. Parlett(1967). "Canonical Decomposition of Hessenberg Matrices," *Math. Comp.* 21, 223–227.

Y. Ikebe(1979). "On Inverses of Hessenberg Matrices," *Lin. Alg. and Its Applic.* 24, 93–97.

尽管 Hessenberg 分解更大程度上被人们看作 QR 分解的 "前期" 分解, 但它在某些问题上相对于较昂贵的 Schur 分解来说较便宜而逐渐流行起来. 在许多应用中已证明它十分有用, 请参阅:

W. Enrisht(1979). "On the Efficient and Reliable Numerical Solution of Large Linear Systems of O.D.E.'s," *IEEE Trans. Auto. Cont* AC-24, 905–908.

G. H. Golub, S. Nash and C. Van Loan(1979). " A Hessenberg-Schur Methold for the Problem *AX+XB=C,*" *IEEE Trans. Auto. Cont. AC*-24, 909–913.

A. Laub(1981). "Efficient Multivariable Frequency Response Computations," *IEEE Trans. Auto. Cont. AC*-26, 407–408.

C. C. Paige(1981). "Properties of Numerical Algorithms Related to Computing Controllability," *IEEE Trans. Auto. Cont. AC*-26, 130–138.

G. Miminis and C. C. Paige(1982). "An Algorithm for Pole Assignment of Time Invariant Linear Systems," *International J. of Control* 35, 341–354.

C. Van Loan(1982). "Using the Hessenberg Decomposition in Control Theory," in *Algorithms and Theory in Filtering and Control,* D. C. Sorensen and R.J.Wets(eds), Mathematical Programming Study No. 18, North Holland, Amsterdam, pp. 102–111.

把多项式求根问题看作矩阵征值问题的可行性之讨论可见:

K. -C. Toh and L. N. Trefethen(1994). "Pseudozeros of Polynomials and Pseudospectra of Companion Matrices," *Numer. Math.* 68, 403–425.

A. Edelman and H. Murakami(1995). "Polynomial Roots from Companion Matrix Eigenvalues," *Math. Comp.* 64, 763–776.

## 7.5　实用 QR 算法

我们重新讨论 Hessenberg QR 迭代, 将它写成:

$$
\begin{aligned}
& H = U_0^{\mathrm{T}} A U_0 \quad (\text{Hessenberg约化}) \\
& \textbf{for } k = 1, 2, \cdots \\
& \qquad H = UR \quad (\text{QR 分解}) \\
& \qquad H = RU \\
& \textbf{end}
\end{aligned}
\tag{7.5.1}
$$

本节的主要目的是描述 $\boldsymbol{H}$ 如何收敛到拟上三角型以及说明为什么 "位移" 可加快收敛速度.

### 7.5.1　降阶

不失一般性, 我们可以假定 (7.5.1) 中每个 Hessenberg 矩阵 $\boldsymbol{H}$ 都是不可约的. 否则的话, 在某一步我们有

$$
\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_{11} & \boldsymbol{H}_{12} \\ 0 & \boldsymbol{H}_{22} \end{bmatrix} \begin{matrix} p \\ n-p \end{matrix}, \qquad \begin{matrix} p & n-p \end{matrix}
$$

其中 $1 \leqslant p < n$, 于是问题通过 "解耦" 变成关于 $\boldsymbol{H}_{11}$ 和 $\boldsymbol{H}_{22}$ 的小问题. 关于这一点, 我们也用 "降阶" 这一术语, 在 $p = n-1$ 或 $p = n-2$ 时常常如此.

实际上, 当 $\boldsymbol{H}$ 的次对角元素适当小的时候, 就可进行解耦. 例如, 在 Eispack 中, 如果

$$
|h_{p+1,p}| \leqslant c\mathbf{u}(|h_{pp}| + |h_{p+1,p+1}|), \tag{7.5.2}
$$

$c$ 为小常数, 就把 $h_{p+1,p}$ "断定" 为零. 这样做是合理的, 因为在整个矩阵中早已有了 $\mathbf{u}\|\boldsymbol{H}\|$ 量级的舍入误差.

### 7.5.2 带位移 QR 迭代

令 $\mu \in \mathbb{R}$, 考虑如下迭代:

$$
\begin{aligned}
& H = U_0^{\mathrm{T}} A U_0 \quad (\text{Hessenberg 约化}) \\
& \textbf{for } k = 1, 2, \cdots \\
& \quad \text{决定标量}\mu \\
& \quad H - \mu I = UR \quad (\text{QR 分解}) \\
& \quad H = RU + \mu I \\
& \textbf{end}
\end{aligned}
\tag{7.5.3}
$$

标量 $\mu$ 称为**位移**. (7.5.3) 中产生的每个矩阵 $\boldsymbol{H}$ 相似于 $\boldsymbol{A}$, 因为

$$\boldsymbol{RU} + \mu \boldsymbol{I} = \boldsymbol{U}^{\mathrm{T}}(\boldsymbol{UR} + \mu \boldsymbol{I})\boldsymbol{U} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{HU}$$

如果我们将 $\boldsymbol{A}$ 的特征值 $\lambda_i$ 排序使得

$$|\lambda_1 - \mu| \geqslant \cdots \geqslant |\lambda_n - \mu|$$

且 $\mu$ 在迭代过程中固定不变, 那么 7.3 节中的理论表明 $\boldsymbol{H}$ 中的第 $p$ 个次对角元素以速度 $\left|\dfrac{\lambda_{p+1} - \mu}{\lambda_p - \mu}\right|^k$ 收敛于 0. 当然, 如果 $\lambda_p = \lambda_{p+1}$, 则根本不收敛. 但是, 比方说, 如果 $\mu$ 比其他特征值更靠近 $\lambda_n$, 那么元素 $(n, n-1)$ 会很快变为零. 极端情形下, 我们有以下定理.

**定理 7.5.1** 令 $\mu$ 是 $n \times n$ 不可约的 Hessenberg 矩阵 $\boldsymbol{H}$ 的特征值, 如果 $\bar{\boldsymbol{H}} = \boldsymbol{RU} + \mu \boldsymbol{I}$, 这里 $\boldsymbol{H} - \mu \boldsymbol{I} = \boldsymbol{UR}$ 是 $\boldsymbol{H} - \mu \boldsymbol{I}$ 的 QR 分解, 那么 $\bar{h}_{n,n-1} = 0$ 且 $\bar{h}_{nn} = \mu$.

**证明** 由于 $\boldsymbol{H}$ 是不可约 Hesserberg 矩阵, 不论 $\mu$ 值为何, $\boldsymbol{H} - \mu \boldsymbol{I}$ 的前 $n-1$ 列是线性无关的. 这样, 如果 $\boldsymbol{UR} = (\boldsymbol{H} - \mu \boldsymbol{I})$ 是 QR 分解, 那么,$r_{ii} \neq 0$, $i = 1 : n-1$. 但是 $\boldsymbol{H} - \mu \boldsymbol{I}$ 奇异, 则 $r_{11} \cdots r_{nn} = 0$. 这样,$r_{nn} = 0$ 且 $\bar{\boldsymbol{H}}(n, :) = [0, \cdots, 0, \mu]$. □

定理表明, 如果用特征值作位移, 一步迭代就能将矩阵降阶.

**例 7.5.1** 如果

$$\boldsymbol{H} = \begin{bmatrix} 9 & -1 & -2 \\ 2 & 6 & -2 \\ 0 & 1 & 5 \end{bmatrix},$$

则 $6 \in \lambda(\boldsymbol{H})$, 如果 $\boldsymbol{UR} = \boldsymbol{H} - 6\boldsymbol{I}$ 是 QR 分解, 则 $\bar{\boldsymbol{H}} = \boldsymbol{RU} + 6\boldsymbol{I}$ 为

$$\bar{\boldsymbol{H}} = \begin{bmatrix} 8.5384 & -3.7313 & -1.0090 \\ 0.6343 & 5.4615 & 1.3867 \\ 0.0000 & 0.0000 & 6.0000 \end{bmatrix}.$$

### 7.5.3 单位移策略

现在让我们考虑, 在迭代过程中随着次对角元素收敛到零, 参考有关 $\lambda(\boldsymbol{A})$ 新的信息而改变 $\mu$ 值. 一种探试法是认为 $h_{nn}$ 为沿对角线的最佳近似特征值. 如果在每次迭代都用此量作为位移, 我们就得到**单位移QR迭代法**:

$$
\begin{aligned}
&\textbf{for}\quad k=1,2,\cdots\\
&\qquad \mu = H(n,n)\\
&\qquad H-\mu I = UR \quad (\text{QR 分解}) \qquad\qquad (7.5.4)\\
&\qquad H = RU+\mu I\\
&\textbf{end}
\end{aligned}
$$

如果 $(n, n-1)$ 元收敛为 0, 则其收敛速度很可能是二次的. 为说明这一点, 我们借用 Stewart(1973,366 页) 的一个例子. 假设 $\boldsymbol{H}$ 是如下不可约的上 Hessenberg 矩阵:

$$
\boldsymbol{H}=\begin{bmatrix}
\times & \times & \times & \times & \times\\
\times & \times & \times & \times & \times\\
0 & \times & \times & \times & \times\\
0 & 0 & \times & \times & \times\\
0 & 0 & 0 & \varepsilon & h_{nn}
\end{bmatrix}
$$

且我们执行一步单位移 QR 算法: $\boldsymbol{UR}=\boldsymbol{H}-h_{nn}\boldsymbol{I}, \bar{\boldsymbol{H}}=\boldsymbol{RU}+h_{nn}\boldsymbol{I}$. 经过化 $\boldsymbol{H}-h_{nn}\boldsymbol{I}$ 为上三角型的 $n-2$ 步后, 我们获得具有如下结构的矩阵:

$$
\boldsymbol{H}=\begin{bmatrix}
\times & \times & \times & \times & \times\\
0 & \times & \times & \times & \times\\
0 & 0 & \times & \times & \times\\
0 & 0 & 0 & a & b\\
0 & 0 & 0 & \varepsilon & 0
\end{bmatrix}
$$

不难证明 $\bar{\boldsymbol{H}}=\boldsymbol{RU}+h_{nn}\boldsymbol{I}$ 的 $(n,n-1)$ 元为 $-\varepsilon^2 b/(\varepsilon^2+a^2)$. 如果我们假定 $\varepsilon \ll a$, 则很清楚新的 $(n,n-1)$ 元之量级为 $\varepsilon^2$, 这恰是我们对一个二次收敛算法所期待的.

**例 7.5.2** 如果

$$
\boldsymbol{H}=\begin{bmatrix}
1 & 2 & 3\\
4 & 5 & 6\\
0 & 0.001 & 7
\end{bmatrix}
$$

且 $\boldsymbol{UR}=\boldsymbol{H}-7\boldsymbol{I}$ 是 QR 分解, 那么 $\bar{\boldsymbol{H}}=\boldsymbol{RU}+7\boldsymbol{I}$ 为

$$
\bar{\boldsymbol{H}}\approx\begin{bmatrix}
-0.5384 & 1.6908 & 0.8351\\
0.3076 & 6.5264 & -6.6555\\
0.0000 & 2\times 10^{-5} & 7.0119
\end{bmatrix}.
$$

像以上的接近最佳位移几乎总能保证 $\bar{h}_{n,n-1}$ 较小. 然而, 这只是一个直观经验, 有例子表明, 即使 $\sigma_{\min}(\boldsymbol{H}-\mu\boldsymbol{I})\approx \mathbf{u}, \bar{h}_{n,n-1}$ 也是一个较大的矩阵元素.

### 7.5.4 双位移策略

不幸的是, 在迭代过程当

$$G = \begin{bmatrix} h_{mm} & h_{mn} \\ h_{nm} & h_{nn} \end{bmatrix}, \quad m = n-1 \tag{7.5.5}$$

的两特征值 $a_1$ 和 $a_2$ 为复数时, 则 $h_{nn}$ 将是一个很坏的近似特征值, 这时,(7.5.4) 有麻烦就是意料之中的.

绕过该困难的途径是, 逐次应用 $a_1$ 和 $a_2$ 作位移量进行两次单位移 QR 迭代:

$$\begin{aligned} \boldsymbol{H} - a_1\boldsymbol{I} &= \boldsymbol{U}_1\boldsymbol{R}_1, \\ \boldsymbol{H}_1 &= \boldsymbol{R}_1\boldsymbol{U}_1 + a_1\boldsymbol{I}, \\ \boldsymbol{H}_1 - a_2\boldsymbol{I} &= \boldsymbol{U}_2\boldsymbol{R}_2, \\ \boldsymbol{H}_2 &= \boldsymbol{R}_2\boldsymbol{U}_2 + a_2\boldsymbol{I}, \end{aligned} \tag{7.5.6}$$

由这些等式可推得

$$(\boldsymbol{U}_1\boldsymbol{U}_2)(\boldsymbol{R}_2\boldsymbol{R}_1) = \boldsymbol{M}, \tag{7.5.7}$$

其中 $\boldsymbol{M}$ 定义为

$$\boldsymbol{M} = (\boldsymbol{H} - a_1\boldsymbol{I})(\boldsymbol{H} - a_2\boldsymbol{I}). \tag{7.5.8}$$

注意到, 即使 $\boldsymbol{G}$ 的特征值为复数, $\boldsymbol{M}$ 也是实矩阵, 这是因为

$$\boldsymbol{M} = \boldsymbol{H}^2 - s\boldsymbol{H} + t\boldsymbol{I},$$

其中 $s = a_1 + a_2 = h_{mm} + h_{nn} = \text{trace}(\boldsymbol{G}) \in \mathbb{R}$, 且

$$t = a_1a_2 = h_{mm}h_{nn} - h_{mn}h_{nm} = \det(\boldsymbol{G}) \in \mathbb{R}.$$

这样,(7.5.7) 是一个实矩阵的 QR 分解且我们可选择 $\boldsymbol{U}_1$ 和 $\boldsymbol{U}_2$ 使得 $\boldsymbol{Z} = \boldsymbol{U}_1\boldsymbol{U}_2$ 是实正交矩阵. 于是,

$$\begin{aligned} \boldsymbol{H}_2 &= \boldsymbol{U}_2^{\mathrm{H}}\boldsymbol{H}_1\boldsymbol{U}_2 = \boldsymbol{U}_2^{\mathrm{H}}(\boldsymbol{U}_1^{\mathrm{H}}\boldsymbol{H}\boldsymbol{U}_1)\boldsymbol{U}_2 \\ &= (\boldsymbol{U}_1\boldsymbol{U}_2)^{\mathrm{H}}\boldsymbol{H}(\boldsymbol{U}_1\boldsymbol{U}_2) = \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z} \end{aligned}$$

是实矩阵.

不幸的是, 舍入误差几乎总是阻止 $\boldsymbol{H}_2$ 回到实数域. 欲保证 $\boldsymbol{H}_2$ 为实矩阵, 则只要:

- 直接给出实矩阵 $\boldsymbol{M} = \boldsymbol{H}^2 - s\boldsymbol{H} + t\boldsymbol{I}$;
- 计算 $\boldsymbol{M} = \boldsymbol{Z}\boldsymbol{R}$ 的实 QR 分解;
- 令 $\boldsymbol{H}_2 = \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}$.

但由于第一步需 $O(n^3)$ 个 flop, 故这并非是实用的.

### 7.5.5 双隐式位移策略

幸运的是, 借助于 7.4.5 节中的隐式 Q 定理, 我们能够只需 $O(n^2)$ 个 flop 就实现双位移步. 确切地说, 如果我们采用以下步骤, 则只需 $O(n^2)$ 个 flop 就能实现从 $\boldsymbol{H}$ 到 $\boldsymbol{H}_2$ 的转变.

- 计算 $\boldsymbol{M}e_1$, 即 $\boldsymbol{M}$ 的第一列.
- 确定 Householder 矩阵 $\boldsymbol{P}_0$ 使得 $\boldsymbol{P}_0(\boldsymbol{M}e_1)$ 是 $e_1$ 的倍数.
- 计算 Householder 矩阵 $\boldsymbol{P}_1, \cdots \boldsymbol{P}_{n-2}$ 使得如果 $\boldsymbol{Z}_1 = \boldsymbol{P}_0\boldsymbol{P}_1 \cdots \boldsymbol{P}_{n-2}$, 那么 $\boldsymbol{Z}_1^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}_1$ 是上 Hessenberg 矩阵且 $\boldsymbol{Z}$ 和 $\boldsymbol{Z}_1$ 的第一列相同.

在这些情况下, 隐式 Q 定理允许我们得出结论, 如果 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}$ 和 $\boldsymbol{Z}_1^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}_1$ 均是不可约的上 Hessenberg 矩阵, 那么它们本质上相等. 注意到如果这些 Hessenberg 矩阵不是不可约的, 那么我们可以进行解耦, 然后就处理较小的不可约子问题.

让我们来进行仔细推导. 首先注意到,$\boldsymbol{P}_0$ 只需 $O(1)$ 个 flop 就能求出, 这是因为 $\boldsymbol{M}e_1 = [x, y, z, 0, \cdots, 0]^{\mathrm{T}}$, 其中

$$
\begin{aligned}
x &= h_{11}^2 + h_{12}h_{21} - sh_{11} + t, \\
y &= h_{21}(h_{11} + h_{22} - s), \\
z &= h_{21}h_{32}.
\end{aligned}
$$

由于相似变换 $\boldsymbol{P}_0$ 只是改变 1,2,3 行和 1,2,3 列, 我们看到

$$
\boldsymbol{P}_0\boldsymbol{H}\boldsymbol{P}_0 = \begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times
\end{bmatrix}
$$

现在 Householder 矩阵 $\boldsymbol{P}_1, \cdots, \boldsymbol{P}_{n-2}$ 之任务是将此矩阵恢复为上 Hessenberg 矩阵. 计算过程如下:

$$
\begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times
\end{bmatrix}
\xrightarrow{\boldsymbol{P}_1}
\begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & 0 & 0 & \times & \times
\end{bmatrix}
\xrightarrow{\boldsymbol{P}_2}
$$

$$
\begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times
\end{bmatrix}
\xrightarrow{\boldsymbol{P}_3}
\begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times
\end{bmatrix}
\xrightarrow{\boldsymbol{P}_4}
$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

很明显, $\boldsymbol{P}_k$ 的形式为 $\boldsymbol{P}_k = \mathrm{diag}(\boldsymbol{I}_k, \bar{\boldsymbol{P}}_k, \boldsymbol{I}_{n-k-3})$, 其中 $\bar{\boldsymbol{P}}_k$ 是 $3\times 3$ 的 Householder 矩阵. 例如:

$$\boldsymbol{P}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

注意到 $\boldsymbol{P}_{n-2}$ 例外, 因为 $\boldsymbol{P}_{n-2} = \mathrm{diag}(\boldsymbol{I}_{n-2}, \bar{\boldsymbol{P}}_{n-2})$.

通过观察 $\boldsymbol{P}_k e_1 = e_1$, $k = 1 : n-2$ 以及 $\boldsymbol{P}_0$ 和 $\boldsymbol{Z}$ 有相同的第一列, 即知可应用定理 7.4.2(隐式 Q 定理). 因此, $\boldsymbol{Z}_1 e_1 = \boldsymbol{Z} e_1$, 且我们能断定, 只要上 Hessenberg 矩阵 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}$ 和 $\boldsymbol{Z}_1^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}_1$ 均不可约, 则 $\boldsymbol{Z}_1$ 和 $\boldsymbol{Z}$ 本质上相等.

从 $H$ 隐式确定 $H_2$ 首先是 Francis(1961) 提出的, 我们称之为 Franics QR步. 完整的 Francis 步归纳如下.

**算法 7.5.1 (Francis QR 步)** 给定不可约的上 Hessenberg 矩阵 $\boldsymbol{H} \in \mathbb{R}^{n\times n}$, 它的最后 $2\times 2$ 主子矩阵有特征值 $a_1$ 和 $a_2$, 本算法计算 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Z}$ 并覆盖 $\boldsymbol{H}$, 这里 $\boldsymbol{Z} = \boldsymbol{P}_1 \cdots \boldsymbol{P}_{n-2}$ 是一列 Householder 矩阵的内积且 $\boldsymbol{Z}^{\mathrm{T}}(\boldsymbol{H} - a_1\boldsymbol{I})(\boldsymbol{H} - a_2\boldsymbol{I})$ 是上三角形矩阵.

$m = n - 1$

{计算 $(H - a_1 I)(H - a_2 I)$ 的第一列}

$s = H(m,m) + H(n,n)$

$t = H(m,m)H(n,n) - H(m,n)H(n,m)$

$x = H(1,1)H(1,1) + H(1,2)H(2,1) - sH(1,1) + t$

$y = H(2,1)(H(1,1) + H(2,2) - s)$

$z = H(2,1)H(3,2)$

**for** $k = 0 : n-3$

$[v, \beta] = \textbf{house}([x \quad y \quad z]^{\mathrm{T}})$

$q = \max\{1, k\}$.

$H(k+1:k+3, q:n) = (I - \beta vv^{\mathrm{T}})H(k+1:k+3, q:n)$

$r = \min\{k+4, n\}$

$H(1:r, k+1:k+3) = H(1:r, k+1:k+3)(I - \beta vv^{\mathrm{T}})$

$x = H(k+2, k+1)$

$y = H(k+3, k+1)$

**if** $k < n-3$

$z = H(k+4, k+1)$

**end**

**end**

$[v, \beta] = \textbf{house}([x \quad y]^{\mathrm{T}})$

$H(n-1:n, n-2:n) = (I - \beta vv^{\mathrm{T}})H(n-1:n, n-2:n)$

$H(1:n, n-1:n) = H(1:n, n-1:n)(I - \beta vv^{\mathrm{T}})$

这一算法需 $10n^2$ 个 flop, 如果把 $Z$ 显式计算成一个正交矩阵, 则还需额外 $10n^2$ 个 flop.

### 7.5.6 完整计算过程

运用算法 7.4.2 约化 $A$ 为 Hessenberg 型, 然后用算法 7.5.1 进行迭代来产生一个实 Schur 型是解决稠密的非对称矩阵特征问题的标准手段. 在迭代过程中必须监视 $H$ 的次对角元素以便发现任何可能的解耦. 下面算法具体演示了如何实现这一点.

**算法 7.5.2(QR 算法)** 给定矩阵 $A \in \mathbb{R}^{n\times n}$ 和比单位舍入误差大的容许误差 tol, 本算法计算实 Schur 标准型 $Q^{\mathrm{T}}AQ = T$. $A$ 用 Hessenberg 分解覆盖. 如果需要求出 $Q$ 和 $T$, 那么 $T$ 储存在 $H$ 中. 如果只是需求特征值, 则 $T$ 的对角块存在 $H$ 中相应的位置.

用算法 7.4.2 来计算 Hessenberg 约化

$\boldsymbol{H} = \boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U}_0$, 其中 $\boldsymbol{U}_0 = \boldsymbol{P}_1\cdots\boldsymbol{P}_{n-2}$.

只要 $Q$ 是所需型 $\boldsymbol{Q} = \boldsymbol{P}_1\cdots\boldsymbol{P}_{n-2}$, 参看 5.1.6 节.

**until** $q = n$

令所有满足

$|h_{i,i-1}| \leqslant \mathrm{tol}(|h_{ii}| + |h_{i-1,i-1}|)$

的次对角元素为 0, 找到最大的非负 $q$ 和最小非负 $p$ 使得

$$H = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ 0 & H_{22} & H_{23} \\ 0 & 0 & H_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$
$$\quad p \quad n-p-q \quad q$$

这里 $H_{33}$ 是拟上三角形矩阵且 $H_{22}$ 是不可约的.(注意: $p$ 或 $q$ 可能为零.)

**if** $q < n$

对 $H_{22}$ 构造一 Francis QR 步: $H_{22} = Z^{\mathrm{T}} H_{22} Z$

**if** 需求 $Q$

$Q = Q \quad \mathrm{diag}(I_p, Z, I_q)$

$H_{22} = H_{12} Z$

$H_{23} = Z^{\mathrm{T}} H_{23}$

**end**

**end**

**end**

将 $\boldsymbol{H}$ 中所有特征值为实的 $2 \times 2$ 的对角块化为上三角形式, 如有必要, 累积变换矩阵.

如果需计算 $\boldsymbol{Q}$ 和 $\boldsymbol{T}$, 此算法需 $25n^3$ 个 flop; 如果只需算特征值, 则需 $10n^3$ 个 flop. 这些 flop 数是很粗的估计, 它们是基于这样的直观经验: 低阶的 $1 \times 1$ 或 $2 \times 2$ 子矩阵解耦, 平均每次仅需两次 Francis 迭代.

**例 7.5.3** 如果算法 7.5.2 应有到

$$\boldsymbol{A} = \begin{bmatrix} 2 & 3 & 4 & 5 & 6 \\ 4 & 4 & 5 & 6 & 7 \\ 0 & 3 & 6 & 7 & 8 \\ 0 & 0 & 2 & 8 & 9 \\ 0 & 0 & 0 & 1 & 10 \end{bmatrix},$$

那么次对角元素有如下收敛 (见下页表).

QR 算法的舍入性质是人们使用任何正交矩阵技术所期望的. 计算得到的实 Schur 型 $\hat{\boldsymbol{T}}$ 正交相似于邻近 $\boldsymbol{A}$ 的矩阵, 即

$$\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{A} + \boldsymbol{E})\boldsymbol{Q} = \hat{\boldsymbol{T}},$$

其中 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} = \boldsymbol{I}$ 且 $\|\boldsymbol{E}\|_2 \approx \mathbf{u}\|\boldsymbol{A}\|_2$. 求得的 $\hat{\boldsymbol{Q}}$ 几乎是正交的, 这是因为 $\hat{\boldsymbol{Q}}^{\mathrm{T}}\hat{\boldsymbol{Q}} = \boldsymbol{I} + \boldsymbol{F}$, 其中 $\|\boldsymbol{F}\|_2 \approx \mathbf{u}$.

$\hat{\boldsymbol{T}}$ 的特征值顺序多少有点任意, 但正如我们在 7.6 节中所述, 利用互换两个相邻对角元素的方法, 就可获得任意排序.

| 迭 代 | $O(\|h_{21}\|)$ | $O(\|h_{32}\|)$ | $O(\|h_{43}\|)$ | $O(\|h_{54}\|)$ |
|---|---|---|---|---|
| 1 | $10^0$ | $10^0$ | $10^0$ | $10^0$ |
| 2 | $10^0$ | $10^0$ | $10^0$ | $10^0$ |
| 3 | $10^0$ | $10^0$ | $10^{-1}$ | $10^0$ |
| 4 | $10^0$ | $10^0$ | $10^{-3}$ | $10^{-3}$ |
| 5 | $10^0$ | $10^0$ | $10^{-6}$ | $10^{-5}$ |
| 6 | $10^{-1}$ | $10^0$ | $10^{-13}$ | $10^{-13}$ |
| 7 | $10^{-1}$ | $10^0$ | $10^{-28}$ | $10^{-13}$ |
| 8 | $10^{-4}$ | $10^0$ | 收敛 | 收敛 |
| 9 | $10^{-8}$ | $10^0$ | | |
| 10 | $10^{-8}$ | $10^0$ | | |
| 11 | $10^{-16}$ | $10^0$ | | |
| 12 | $10^{-32}$ | $10^0$ | | |
| 13 | 收敛 | 收敛 | | |

### 7.5.7 平衡

最后, 我们应注意到, 如果 $\boldsymbol{A}$ 的元素的数量级变化很大, 则在应用 QR 算法之前, 应对 $\boldsymbol{A}$ 进行平衡. 这需要 $O(n^2)$ 次运算来计算对角矩阵 $\boldsymbol{D}$ 使得若

$$\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D} = [\boldsymbol{c}_1 \ \cdots \ \boldsymbol{c}_n] = \begin{bmatrix} \boldsymbol{r}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{r}_n^{\mathrm{T}} \end{bmatrix}$$

则 $\|\boldsymbol{r}_i\|_\infty \approx \|\boldsymbol{c}_i\|_\infty$, $i = 1:n$. 对角矩阵 $\boldsymbol{D}$ 选成具有形式 $\boldsymbol{D} = \mathrm{diag}(\beta^{i_1}, \cdots, \beta^{i_n})$, 其中 $\beta$ 是浮点基数. 注意, 这样计算 $\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}$ 就可以没有舍入误差. 当 $\boldsymbol{A}$ 被平衡后, 计算的特征值常常会更精确. 参看 Parlett and Reinsch(1969).

## 习 题

**7.5.1** 证明: 若 $\bar{\boldsymbol{H}} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{Q}$ 是用 $\boldsymbol{H} = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$ 执行一单位移 QR 步而获得的, 则 $|\bar{h}_{21}| \leqslant |y^2x|/[(w-z)^2 + y^2]$.

**7.5.2** 给出求 $2 \times 2$ 对角矩阵 $\boldsymbol{D}$ 的公式, 这里 $\boldsymbol{D}$ 使得 $\|\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}\|_F$ 极小化, 其中 $\boldsymbol{A} = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$.

**7.5.3** 试解释单位移 QR 步 $\boldsymbol{H} - \mu\boldsymbol{I} = \boldsymbol{U}\boldsymbol{R}, \bar{\boldsymbol{H}} = \boldsymbol{R}\boldsymbol{U} + \mu\boldsymbol{I}$ 是怎样隐式执行的. 即表明不用从 $\boldsymbol{H}$ 的对角线减去 $\mu$, 从矩阵 $\bar{H}$ 到 $H$ 的变换是怎样进行的.

**7.5.4** 设 $\boldsymbol{H}$ 是上 Hessenberg 矩阵且我们用列选主元的高斯消去法计算分解 $\boldsymbol{P}\boldsymbol{H} = \boldsymbol{L}\boldsymbol{U}$. 证明: $\boldsymbol{H}_1 = \boldsymbol{U}(\boldsymbol{P}^{\mathrm{T}}\boldsymbol{L})$ 是上 Hessenberg 矩阵且相似于 $\boldsymbol{H}$. (这是修正 LR 算法的基础.)

**7.5.5** 证明: 若 $\boldsymbol{H} = \boldsymbol{H}_0$ 给定且我们由 $\boldsymbol{H}_k - \mu_k\boldsymbol{I} = \boldsymbol{U}_k\boldsymbol{R}_k$, $\boldsymbol{H}_{k+1} = \boldsymbol{R}_k\boldsymbol{U}_k + \mu_k\boldsymbol{I}$ 产生矩阵 $\boldsymbol{H}_k$, 则

$$(\boldsymbol{U}_1 \cdots \boldsymbol{U}_j)(\boldsymbol{R}_j \cdots \boldsymbol{R}_1) = (\boldsymbol{H} - \mu_1\boldsymbol{I}) \cdots (\boldsymbol{H} - \mu_j\boldsymbol{I}).$$

## 本节注释与参考文献

实用 QR 算法的发展起始于下面这篇重要论文:

H. Rutishauser(1958). "Solution of Eigenvalue Problems with the LR Transformation," *Nat. Bur. Stand. App. Math. Ser.* 49, 47–81.

然后, 上文描述的算法被 "正交化", 见:

J. G. F. Francis(1961). "The QR Transformation:A Unitary Analogue to the LR Transformation, Parts I and II" *Comp. J.* 4, 265–272, 332–345.

关于实用 QR 算法的论述可见 Wilkinson(1965), Stewart(1973) 以及 Watkins(1991), 也可见:

D. Watkins and L. Elsner(1991). " Chasing Algorithms for the Eigenvalue Problem," *SIAM J. Matrix Anal. Appl.* 12, 374–384.

D. S. Watkins and L. Elsner(1991). " Convergence of Algorithms of Decomposition Type for the Eigenvalue Problem," *Lin. Alg. and Its Application* 143, 19–47.

J. Erxiong(1992). "A Note on the Double-Shift QL Algorithm," *Lin. Alg. and Its Application* 171, 121–132.

LR 和 QR 方法的 Algol 程序见:

R. S. Martin and J. H. Wilkinson(1968). "The Modified LR Algorithm for Complex Hessenberg Matrices, " *Numer. Math.* 12, 369–376. 也可见 Wilkinson and Reinsch(1971, 369–403 页).

R. S. Martin, G. Peters, and J. H. Wilkinson(1970). "The QR Algorithm for Real Hessenberg Matrices," *Numer. Math.* 14, 219–231. 也可见 Wilkinson and Reinsch(1971, 359–371 页).

有关平衡问题方面在下面文章中讨论到:

E. E. Osborne(1960). "On Preconditioning of Matrices," *JACM* 7, 338–345.

B. N. Parlett and C. Reinsch(1969). "Balancing a Matrix for Calculation of Eigenvalues and Eigenvectors, " *Numer. Math.* 13, 292–304. 也可见 Wilkinson and Reinsch(1971, 315–326 页).

高性能特征值求解的论文包括:

Z. Bai and J. W. Demmel(1989). "On a Block Implementation of Hessenberg Multishift QR Iteration, " *Int'l J. of High Speed Comput.* 1, 97–112.

G. Shroff(1991). "A Parallel Algorithm for the Eigenvalues and Eigenvectors of a General Complex Matrix," *Numer. Math.* 58, 779–806.

R. A. Van De Geijn (1993). "Deferred Shifting Schemes for Parallel QR Methods," *SIAM J. Matrix Anal. Appl.* 14, 180–194.

A. A. Dubrulle and G. H. Golub(1994). "A Multishift QR Iteration Without Computation of the Shifts, " *Numerical Algorithms* 7, 173–181.

## 7.6 不变子空间计算

一旦实 Schur 分解 $Q^{\mathrm{T}}AQ = T$ 已算出, 几个重要的不变子空间问题就能解决. 本节我们讨论下列问题:

- 计算与 $\lambda(A)$ 的某个子集所对应的特征向量;
- 计算给定不变子空间的标准正交基;
- 用良态相似变换将 $A$ 块对角化;
- 计算一组特征向量基 (不管它们的条件);
- 计算 $A$ 的相似 Jordan 标准型.

稀疏矩阵的特征向量和不变子空间的计算在别处讨论. 参看 7.3 节以及第 8 章和第 9 章的部分内容.

### 7.6.1 由逆迭代计算选定的特征向量

令 $q^{(0)} \in \mathbb{C}^n$ 是给定的单位 2 范数向量并设 $A - \mu I \in \mathbb{R}^{n\times n}$ 非奇异. 下面算法称为逆迭代:

$$
\begin{aligned}
&\textbf{for}\quad k = 1, 2, \cdots \\
&\qquad \text{解 } (A - \mu I)z^{(k)} = q^{(k-1)} \\
&\qquad q^{(k)} = z^{(k)}/||z^k||_2 \\
&\qquad \lambda^{(k)} = q^{(k)\mathrm{T}} A q^{(k)} \\
&\textbf{end}
\end{aligned}
\tag{7.6.1}
$$

逆迭代就是应用到 $(A - \mu I)^{-1}$ 上的幂法.

为了分析 (7.6.1) 的表现, 设 $A$ 有一组特征向量基 $\{x_1, \cdots, x_n\}$ 且对 $i = 1 : n$, $Ax_i = \lambda_i x_i$. 如果

$$q^{(0)} = \sum_{i=1}^{n} \beta_i x_i,$$

则 $q^{(k)}$ 是单位向量, 其方向为

$$(A - \mu I)^{-k} q^{(0)} = \sum_{i=1}^{n} \frac{\beta_i}{(\lambda_i - \mu)^k} x_i.$$

显然, 如果 $\mu$ 比其他特征值更接近 $\lambda_j$, 则只要 $\beta_j \neq 0$, $q^{(k)}$ 在 $x_j$ 方向的分量就非常多.

(7.6.1) 的终止准则的例子是只要余量 $r^{(k)} = (A - \mu I)q^{(k)}$ 满足

$$||r^{(k)}||_\infty \leqslant c\mathbf{u}||A||_\infty \tag{7.6.2}$$

就终止, 其中 $c$ 是数量级为 1 的常数. 由于

$$(A + E_k)q^{(k)} = \mu q^{(k)},$$

其中 $E_k = -r^{(k)} q^{(k)^{\mathrm{T}}}$, 可知 (7.6.2) 使得 $\mu$ 和 $q^{(k)}$ 是邻近矩阵的精确特征对.

逆迭代可以与 QR 算法一起用:

- 计算 Hessenberg 分解 $U_0^{\mathrm{T}}AU_0 = H$;
- 应用隐式双位移 Francis 迭代于 $H$(不用累积变换矩阵);
- 对每个已求得的特征值 $\lambda$, 欲找相应特征向量 $x$, 应用 (7.6.1), 令 $A = H$, $\mu = \lambda$, 产生一个向量 $z$, 使得 $Hz \approx \mu z$;
- 令 $x = U_0 z$.

$H$ 的逆迭代是很经济的, 这是因为: (1) 在双 Francis 迭代过程中我们不必计算累积变换矩阵; (2) 只需 $O(n^2)$ 个 flop 就能得到形如 $H - \lambda I$ 的因子矩阵;(3) 一般只需一次迭代就能产生一个足够近似的特征向量.

这最后一点也许是逆迭代最有趣的方面, 且需要作些验证, 因为如果 $\lambda$ 是病态的, 它可能相当不精确. 为简单起见, 设 $\lambda$ 为实数且令

$$H - \lambda I = \sum_{i=1}^{n} \sigma_i u_i v_i^{\mathrm{T}} = U\Sigma V^{\mathrm{T}}$$

是 $H - \lambda I$ 的 SVD, 从 7.5.6 节所讲过的 QR 算法的舍入性质知, 存在一个矩阵 $E \in \mathbb{R}^{n\times n}$ 使得 $H + E - \lambda I$ 奇异且 $\|E\|_2 \approx \mathbf{u}\|H\|_2$. 可以得出 $\sigma_n \approx \mathbf{u}\sigma_1$ 且 $\|(H - \hat{\lambda} I)v_n\|_2 \approx \mathbf{u}\sigma_1$, 即 $v_n$ 是好的近似特征向量. 很显然, 如果初始向量 $q^{(0)}$ 有展式

$$q^{(0)} = \sum_{i=1}^{n} \gamma_i u_i,$$

则 $z^{(1)} = \sum_{i=1}^{n} \frac{\gamma_i}{\sigma_i} v_i$ 就有 "很多" $v_n$ 方向的分量. 注意, 如果 $s(\lambda) \approx |u_n^{\mathrm{T}} v_n|$ 小, 则 $z^{(1)}$ 在 $u_n$ 方向上严重缺少. 这解释了 (凭经验) 为什么再来一步逆迭代不大可能产生更好的近似特征向量, 当 $\lambda$ 是病态的时尤为如此. 欲知详情, 请看 Peters and Wilkinson(1979).

**例 7.6.1** 矩阵

$$A = \begin{bmatrix} 1 & 1 \\ 10^{-10} & 1 \end{bmatrix}$$

有特征值 $\lambda_1 = 0.999\,99$ 和 $\lambda_2 = 1.000\,01$, 且相应的特征向量为 $x_1 = [1, -10^{-5}]^{\mathrm{T}}$, $x_2 = [1, 10^{-5}]^{\mathrm{T}}$. 两个特征值的条件数的数量级均为 $10^5$, 近似特征值 $\mu = 1$ 恰 $A + E$ 的特征值, 其中

$$E = \begin{bmatrix} 0 & 0 \\ -10^{-10} & 0 \end{bmatrix}.$$

这样, 当用 10 位浮点数运算时, $\mu$ 的精度是用 QR 算法计算特征值典型的精度.

如果 (7.6.1) 中用初始向量 $q^{(0)} = [0, 1]^{\mathrm{T}}$, 则 $q^{(1)} = [1, 0]^{\mathrm{T}}$ 且 $\|Aq^{(1)} - \mu q^{(1)}\|_2 = 10^{-10}$. 然而, 再算一步却产生 $q^{(2)} = [0, 1]^{\mathrm{T}}$, 其中 $\|Aq^{(2)} - \mu q^{(2)}\|_2 = 1$. 此例在 Peters and Wilkinson(1979) 中讨论过.

### 7.6.2 在实 Schur 型中对特征值进行排序

再次指出, 实 Schur 分解给出不变子空间的信息. 如果

$$Q^{\mathrm{T}}AQ = T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{matrix} p \\ q \end{matrix}$$
$$\qquad\qquad\qquad\qquad p \quad\; q$$

且 $\lambda(T_{11}) \cap \lambda(T_{22}) = \varnothing$ , 那么 $Q$ 的前 $p$ 列张成一个与 $\lambda(T_{11})$ 相对应的唯一不变子空间.(见 7.1.4 节.) 可惜, Francis 迭代只是给了我们一个在 $T_F$ 的对角线上特征值随机出现的实 Schur 分解 $Q_F^{\mathrm{T}}AQ_F = T_F$. 如果我们需求一个不变子空间的正交基, 而相关特征值都不在 $T_F$ 的对角线的上部, 则会产生问题. 很明显, 我们需要一种方法计算正交矩阵 $Q_D$, 使得 $Q_D^{\mathrm{T}}T_FQ_D$ 是特征值按适当顺序排列的拟上三角形矩阵.

看看 $2\times 2$ 的情形, 我们就能明白如何实现这一点. 设

$$Q_F^{\mathrm{T}}AQ_F = T_F = \begin{bmatrix} \lambda_1 & t_{12} \\ 0 & \lambda_2 \end{bmatrix}, \quad \lambda_1 \neq \lambda_2,$$

且我们希望颠倒特征值的顺序. 注意 $T_Fx = \lambda_2 x$, 其中

$$x = \begin{bmatrix} t_{12} \\ \lambda_2 - \lambda_1 \end{bmatrix}.$$

令 $Q_D$ 是 Givens 旋转变换, 使得 $Q_D^{\mathrm{T}}x$ 的第二个分量为零. 如果 $Q = Q_FQ_D$, 则

$$(Q^{\mathrm{T}}AQ)e_1 = Q_D^{\mathrm{T}}T_F(Q_De_1) = \lambda_2 Q_D^{\mathrm{T}}(Q_De_1) = \lambda_2 e_1.$$

于是 $Q^{\mathrm{T}}AQ$ 一定具有如下形式:

$$Q^{\mathrm{T}}AQ = \begin{bmatrix} \lambda_2 & \pm t_{12} \\ 0 & \lambda_1 \end{bmatrix}.$$

假定在对角线上没遇到 $2\times 2$ 块, 则通过应用这一技巧系统地互换相邻特征值, 我们就能把 $\lambda(A)$ 的任意子集移到 $T$ 的对角线最上面.

**算法 7.6.1** 给定正交矩阵 $Q \in \mathbb{R}^{n\times n}$, 上三角形矩阵 $T = Q^{\mathrm{T}}AQ$,$\lambda(A)$ 的子集 $\Delta = \{\lambda_1, \cdots, \lambda_p\}$, 本算法计算一个正交矩阵 $Q_D$ 使得 $Q_D^{\mathrm{T}}TQ_D = S$ 为上三角形矩阵且 $\{s_{11}, \cdots, s_{pp}\} = \Delta$, 矩阵 $Q$ 和 $T$ 分别被 $QQ_D$ 和 $S$ 覆盖.

**while** $\{t_{11}, \cdots, t_{pp}\} \neq \Delta$
  **for** $k = 1 : n-1$
    **if** $t_{kk} \notin \Delta$ 且 $t_{k+1,k+1} \in \Delta$
     $[c, s] = \textbf{givens}(T(k, k+1), T(k+1, k+1) - T(k, k))$
     $T(k:k+1, k:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} T(k:k+1, k:n)$

$$T(1:k+1,k:k+1)=T(1:k+1,k:k+1)\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

$$\boldsymbol{Q}(1:n,k:k+1)=\boldsymbol{Q}(1:n,k:k+1)\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

**end**

**end**

**end**

这算法需 $k(12n)$ 个 flop, 这里 $k$ 是所需交换的总次数. 整数 $k$ 决不大于 $(n-p)p$.

当 $\boldsymbol{T}$ 的对角线上有 $2\times 2$ 块时, 交换变得稍为复杂. 欲知详情, 请看 Ruhe(1970) 和 Stewart(1976). 当然, 这些交换技巧可用于特征值排序, 比方说按模从大到小排列.

通过进行实 Schur 分解来计算不变子空间是非常稳定的. 如果记 $\hat{\boldsymbol{Q}}=[\hat{q}_1,\cdots,\hat{q}_n]$ 为所计算的正交矩阵 $\boldsymbol{Q}$, 则 $\|\hat{\boldsymbol{Q}}^{\mathrm{T}}\hat{\boldsymbol{Q}}-\boldsymbol{I}\|_2\approx\mathbf{u}$, 且存在矩阵 $\boldsymbol{E}$ 满足 $\|\boldsymbol{E}\|_2\approx\mathbf{u}\|\boldsymbol{A}\|_2$, 使得对 $i=1:p$ 有 $(\boldsymbol{A}+\boldsymbol{E})\hat{q}_i\in\operatorname{span}\{\hat{q}_1,\cdots,\hat{q}_p\}$.

### 7.6.3 块对角化

令

$$\boldsymbol{T}=\begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} & \cdots & \boldsymbol{T}_{1_q} \\ \mathbf{0} & \boldsymbol{T}_{22} & \cdots & \boldsymbol{T}_{2_q} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{T}_{qq} \end{bmatrix}\begin{matrix} n_1 \\ n_2 \\ \\ n_q \end{matrix} \qquad (7.6.3)$$
$$\qquad\quad n_1 \quad\; n_2 \qquad\quad n_q$$

是某实 Schur 标准型 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}\in\mathbb{R}^{n\times n}$ 的划分, 使得 $\lambda(\boldsymbol{T}_{11}),\cdots,\lambda(\boldsymbol{T}_{qq})$ 不相交. 由定理 7.1.6, 存在矩阵 $\boldsymbol{Y}$ 使得 $\boldsymbol{Y}^{-1}\boldsymbol{T}\boldsymbol{Y}=\operatorname{diag}(\boldsymbol{T}_{11},\cdots,\boldsymbol{T}_{qq})$. 现在, 我们给出计算 $\boldsymbol{Y}$ 的实际算法以及 $\boldsymbol{Y}$ 的敏感度 (作为以上划分的函数) 分析.

将 $\boldsymbol{I}_n=[\boldsymbol{E}_1,\cdots,\boldsymbol{E}_q]$ 与 $\boldsymbol{T}$ 相符划分且定义 $\boldsymbol{Y}_{ij}\in\mathbb{R}^{n\times n}$ 如下:

$$\boldsymbol{Y}_{ij}=\boldsymbol{I}_n+\boldsymbol{E}_i\boldsymbol{Z}_{ij}\boldsymbol{E}_j^{\mathrm{T}},\quad i<j,\boldsymbol{Z}_{ij}\in\mathbb{R}^{n_i\times n_j}.$$

换句话说, 除 $\boldsymbol{Z}_{ij}$ 取代 $(i,j)$ 块位置外, $\boldsymbol{Y}_{ij}$ 看上去就是单位矩阵. 由此推出, 如果 $\boldsymbol{Y}_{ij}^{-1}\boldsymbol{T}\boldsymbol{Y}_{ij}=\bar{\boldsymbol{T}}=(\bar{\boldsymbol{T}}_{ij})$, 则 $\boldsymbol{T}$ 和 $\bar{\boldsymbol{T}}$ 除了

$$\begin{aligned} \bar{\boldsymbol{T}}_{ij}&=\boldsymbol{T}_{ii}\boldsymbol{Z}_{ij}-\boldsymbol{Z}_{ij}\boldsymbol{T}_{jj}+\boldsymbol{T}_{ij}, \\ \bar{\boldsymbol{T}}_{ik}&=\boldsymbol{T}_{ik}-\boldsymbol{Z}_{ij}\boldsymbol{T}_{jk}\quad(k=j+1:q), \\ \bar{\boldsymbol{T}}_{kj}&=\boldsymbol{T}_{ki}\boldsymbol{Z}_{ij}+\boldsymbol{T}_{kj}\quad(k=1:i-1). \end{aligned}$$

之外都是一样的. 这样, 只要我们有算法可求解 Sylvester方程

$$\boldsymbol{F}\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{G}=\boldsymbol{C}, \qquad (7.6.4)$$

其中 $\boldsymbol{F}\in\mathbb{R}^{p\times p},\boldsymbol{G}\in\mathbb{R}^{r\times r}$ 是给定的拟上三角形矩阵,$\boldsymbol{C}\in\mathbb{R}^{p\times r}$, 则 $\boldsymbol{T}_{ij}$ 能被化为零.

Bartels and Stewart(1972) 设计了触决此问题之方法. 令 $\boldsymbol{C}=[\boldsymbol{c}_1\ \cdots\ \boldsymbol{c}_r]$ 和 $\boldsymbol{Z}=[\boldsymbol{z}_1\ \cdots\ \boldsymbol{z}_r]$ 按列划分. 如果 $g_{k+1,k}=0$, 则通过比较 (7.6.4) 中的列我们有

$$\boldsymbol{F}\boldsymbol{z}_k-\sum_{i=1}^{k}g_{ik}\boldsymbol{z}_i=\boldsymbol{c}_k.$$

这样, 一旦我们知道 $\boldsymbol{z}_1,\cdots,\boldsymbol{z}_{k-1}$, 就可解拟三角形方程组

$$(\boldsymbol{F}-g_{kk}\boldsymbol{I})\boldsymbol{z}_k=\boldsymbol{c}_k+\sum_{i=1}^{k-1}g_{ik}\boldsymbol{z}_i$$

得到 $\boldsymbol{z}_k$. 如果 $g_{k+1,k}\neq 0$, 则通过解 $2p\times 2p$ 方程组

$$\begin{bmatrix}\boldsymbol{F}-g_{kk}\boldsymbol{I} & -g_{mk}\boldsymbol{I}\\ -g_{km}\boldsymbol{I} & \boldsymbol{F}-g_{mm}\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{z}_k\\ \boldsymbol{z}_m\end{bmatrix}=\begin{bmatrix}\boldsymbol{c}_k\\ \boldsymbol{c}_m\end{bmatrix}+\sum_{i=1}^{k-1}\begin{bmatrix}g_{ik}\boldsymbol{z}_i\\ g_{im}\boldsymbol{z}_i\end{bmatrix}\tag{7.6.5}$$

同时求得 $\boldsymbol{z}_k$ 和 $\boldsymbol{z}_{k+1}$, 上式中 $m=k+1$. 按照排列 $(1,p+1,2,p+2,\cdots,p,2p)$ 重组这些方程, 就可获得一个用 $O(p^2)$ 个 flop 就可求解的带状方程组. 可在 Bartels and Stewart(1972) 中找到具体细节. 以下是当 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 均为三角形矩阵时完整的算法过程.

**算法 7.6.2(Bartels-Stewart 算法)** 给定 $\boldsymbol{C}\in\mathbb{R}^{p\times r}$, 上三角形矩阵 $\boldsymbol{F}\in\mathbb{R}^{p\times p}$ 和 $\boldsymbol{G}\in\mathbb{R}^{r\times r}$, 满足 $\lambda(\boldsymbol{F})\cap\lambda(\boldsymbol{G})=\varnothing$, 本算法用方程 $\boldsymbol{FZ}-\boldsymbol{ZG}=\boldsymbol{C}$ 的解覆盖 $\boldsymbol{C}$.

```
for k = 1 : r
    C(1 : p, k) = C(1 : p, k) + C(1 : p, 1 : k − 1)G(1 : k − 1, k)
    从(F − G(k, k)I)z = C(1 : p, k)解出z
    C(1 : p, k) = z
end
```

此算法需要 $pr(p+r)$ 个 flop.

将 $\boldsymbol{T}$ 的严格上三角块 (super diagonal block) 按适当的顺序化为零, 整个矩阵就能约化为分块对角型.

**算法 7.6.3** 给定一个正交矩阵 $\boldsymbol{Q}\in\mathbb{R}^{n\times n}$, 一个拟上三角形矩阵 $\boldsymbol{T}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$, 且有 (7.6.3) 的分块, 下面的算法用 $\boldsymbol{QY}$ 覆盖 $\boldsymbol{Q}$, 其中 $\boldsymbol{Y}^{-1}\boldsymbol{TY}=\mathrm{diag}(\boldsymbol{T}_{11},\cdots,\boldsymbol{T}_{qq})$.

```
for j = 2 : q
    for i = 1 : j − 1
        用算法 7.6.2 解T_ii Z − Z T_jj = −T_ij 求出Z
        for k = j + 1 : q
            T_ik = T_ik − Z T_jk
        end
        for k = 1 : q
            Q_kj = Q_ki Z + Q_kj
        end
    end
end
```

该方法所需的 flop 数是 (7.6.3) 中分块尺寸的较为复杂的函数.

实 Schur 型 $\boldsymbol{T}$ 及其在 (7.6.3) 中划分的选择, 决定了在算法 7.6.3 中必须求解的 Sylvester 方程的灵敏性. 这就影响矩阵 $\boldsymbol{Y}$ 的条件数和分块对角化的整个有效性. 有这些依赖关系的原因是方程

$$\boldsymbol{T}_{ii}\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{T}_{jj} = -\boldsymbol{T}_{ij} \tag{7.6.6}$$

的计算解 $\hat{Z}$ 的相对误差满足

$$\frac{\|\hat{\boldsymbol{Z}} - \boldsymbol{Z}\|_F}{\|\boldsymbol{Z}\|_F} \approx \mathbf{u}\frac{\|\boldsymbol{T}\|_F}{\operatorname{sep}(\boldsymbol{T}_{ii}, \boldsymbol{T}_{jj})}.$$

欲知详情, 请看 Golub, Nash and Van Loan(1979). 因为

$$\operatorname{sep}(\boldsymbol{T}_{ii}, \boldsymbol{T}_{jj}) = \min_{X \neq 0} \frac{\|\boldsymbol{T}_{ii}\boldsymbol{X} - \boldsymbol{X}\boldsymbol{T}_{jj}\|_F}{\|\boldsymbol{X}\|_F} \leqslant \min_{\substack{\lambda \in \lambda(\boldsymbol{T}_{ii}) \\ \mu \in \lambda(\boldsymbol{T}_{jj})}} |\lambda - \mu|,$$

所以只要子集 $\lambda(\boldsymbol{T}_{ii})$ 没被足够分离, 精度就可能严重损失. 如 $Z$ 满足 (7.6.6), 则

$$\|\boldsymbol{Z}\|_F \leqslant \frac{\|\boldsymbol{T}_{ij}\|_F}{\operatorname{sep}(\boldsymbol{T}_{ii}, \boldsymbol{T}_{jj})}.$$

这样, 如果 $\operatorname{sep}(\boldsymbol{T}_{ii}, \boldsymbol{T}_{jj})$ 小, 那么大范数解将是意料之中的. 这就使得在算法 7.6.3中矩阵 $\boldsymbol{Y}$ 变成病态, 因为 $\boldsymbol{Y}$ 是矩阵 $\boldsymbol{Y}_{ij} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{Z} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}$ 之积. 注意: $\kappa_F(\boldsymbol{Y}_{ij}) = 2n + \|\boldsymbol{Z}\|_F^2$.

针对这些困难, Bavely and Stewart(1979) 提出了一个分块对角化的算法, 它动态地决定 (7.6.3) 中的特征值顺序和分块, 使得算法 7.6.3 中所有 $\boldsymbol{Z}$ 矩阵之范数不超过用户给定的容限. 他们发现通过控制 $\boldsymbol{Y}_{ij}$ 的条件数可以控制 $\boldsymbol{Y}$ 的条件数.

### 7.6.4 特征向量基

如果划分 (7.6.3) 中的每一块都是 $1 \times 1$ 块, 则算法 7.6.3 产生一组特征向量基. 与逆迭代法一样, 计算出的特征值和特征向量对正好相对某一 “邻近” 矩阵是精确的. 广泛采用以下规则来确定适当的特征向量方法: 每当想要的特征向量少于 25% 时, 就使用逆迭代.

但是, 我们需指出实 Schur 型也可用来计算指定的特征向量. 设

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{T}_{11} & u & \boldsymbol{T}_{13} \\ 0 & \lambda & v^{\mathrm{T}} \\ 0 & 0 & \boldsymbol{T}_{33} \end{bmatrix} \begin{matrix} k-1 \\ 1 \\ n-k \end{matrix}$$
$$\qquad\qquad k-1 \quad 1 \quad n-k$$

是上拟三角形矩阵且 $\lambda \notin \lambda(\boldsymbol{T}_{11}) \cup \lambda(\boldsymbol{T}_{33})$. 由此可见, 如果我们解线性方程组 $(\boldsymbol{T}_{11} - \lambda\boldsymbol{I})\boldsymbol{w} = -\boldsymbol{u}$ 和 $(\boldsymbol{T}_{33} - \lambda\boldsymbol{I})^{\mathrm{T}}\boldsymbol{z} = -\boldsymbol{v}$, 则

$$x = Q\begin{bmatrix} w \\ 1 \\ 0 \end{bmatrix} \text{ 和 } y = Q\begin{bmatrix} 0 \\ 1 \\ z \end{bmatrix}$$

是相应的右特征向量和左特征向量. 注意: $\lambda$ 的条件数为

$$1/s(\lambda) = \sqrt{(1 + w^{\mathrm{T}}w)(1 + z^{\mathrm{T}}z)}.$$

### 7.6.5 确定 Jordan 块结构

设我们已求出实 Schur 分解 $A = QTQ^{\mathrm{T}}$, 已确定了 “相等” 特征值的组且计算了相应的块对角化 $T = Y\mathrm{diag}(T_{11}, \cdots, T_{qq})Y^{-1}$. 正如我们已讨论的, 这是一个艰巨的任务. 然而, 如果我们试图去确定每个 $T_{ii}$ 的 Jordan 块结构, 就会碰到更大的数值问题. 简要考察这些难点是为了说明 Jordan 分解的局限性.

为清楚起见, 设 $\lambda(T_{ii})$ 为实数. 约化 $T_{ii}$ 为 Jordan 型, 先用形如 $C = \lambda I + N$ 的矩阵来代替 $T_{ii}$, 其中 $N$ 是 $T_{ii}$ 的严格上三角部分, $\lambda$ 为其特征值的平均值.

回想到 Jordan 块 $J(\lambda)$ 的维数是使 $[J(\lambda) - \lambda I]^k = 0$ 的最小非负整数. 这样, 如果 $p_i = \dim[\mathrm{null}(N^i)]$, $i = 0 : n$, 则 $p_i - p_{i-1}$ 等于 $C$ 中维数大于或等于 $i$ 的 Jordan 块的数目. 一个具体例子有助于弄清楚这一断言和说明 SVD 在 Jordan 型计算中的作用.

设 $C$ 是 $7 \times 7$ 矩阵. 假设我们计算 SVD 为 $U_1^{\mathrm{T}}NV_1 = \Sigma_1$ 且 “发现” $N$ 的秩为 3. 如果我们将奇异值按从小到大排列, 则可得出矩阵 $N_1 = V_1^{\mathrm{T}}NV_1$ 形如

$$N_1 = \begin{bmatrix} \mathbf{0} & K \\ \mathbf{0} & L \end{bmatrix} \begin{matrix} 4 \\ 3 \end{matrix}$$
$$\begin{matrix} 4 & 3 \end{matrix}$$

于是, $\lambda$ 的几何重数为 4, 即 $C$ 的 Jordan 型有 4 块 $(p_1 - p_0 = 4 - 0 = 4)$.

现在设 $\tilde{U}_2^{\mathrm{T}}L\tilde{V}_2 = \Sigma_2$ 是 $L$ 的 SVD 且我们发现 $L$ 的秩为 1. 若我们又一次将奇异值按从小到大排, 则 $L_2 = \tilde{V}_2^{\mathrm{T}}L\tilde{V}_2$ 很明显有如下结构:

$$L_2 = \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ 0 & 0 & c \end{bmatrix}.$$

然而 $\lambda(L_2) = \lambda(L) = \{0, 0, 0\}$, 这样 $c = 0$. 所以, 如果

$$V_2 = \mathrm{diag}(I_4, \tilde{V}_2),$$

则 $N_2 = V_2^{\mathrm{T}}N_1V_2$ 具有形式:

$$N_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & a \\ 0 & 0 & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

除了允许我们在上三角块产生更多零外, $\boldsymbol{L}$ 的 SVD 也使我们能推出 $\boldsymbol{N}^2$ 的零空间维数. 由于

$$\boldsymbol{N}_1^2 = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{KL} \\ \boldsymbol{0} & \boldsymbol{L}^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{K} \\ \boldsymbol{0} & \boldsymbol{L} \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{K} \\ \boldsymbol{0} & \boldsymbol{L} \end{bmatrix},$$

且 $\begin{bmatrix} \boldsymbol{K} \\ \boldsymbol{L} \end{bmatrix}$ 是列满秩的,

$$p_2 = \dim(\text{null}(N^2)) = \dim(\text{null}(\boldsymbol{N}_1^2)) = 4 + \dim(\text{null}(\boldsymbol{L})) = p_1 + 2.$$

这样, 到此我们能断定 $\boldsymbol{C}$ 的 Jordan 型至少有两个维数大于或等于 2 的块.

最后, 容易看出 $\boldsymbol{N}_1^3 = 0$, 由此可推得有 $p_3 - p_2 = 7 - 6 = 1$ 个维数大于或等于 3 的块. 若我们定义 $\boldsymbol{V} = \boldsymbol{V}_1\boldsymbol{V}_2$, 则可得分解

$$\boldsymbol{V}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{V} = \begin{bmatrix} \lambda & 0 & 0 & 0 & \times & \times & \times \\ 0 & \lambda & 0 & 0 & \times & \times & \times \\ 0 & 0 & \lambda & 0 & \times & \times & \times \\ 0 & 0 & 0 & \lambda & \times & \times & \times \\ 0 & 0 & 0 & 0 & \lambda & \times & a \\ 0 & 0 & 0 & 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\\a\end{matrix}}\right\} \text{4块阶为 1 或更大} \\ \left.\vphantom{\begin{matrix}a\\a\end{matrix}}\right\} \text{2块阶为 2 或更大} \\ \left.\vphantom{a}\right\} \text{1块阶为 3 或更大} \end{matrix}$$

"显示"$\boldsymbol{C}$ 的 Jordan 块结构: 2 块阶为 1, 1 块阶为 2, 1 块阶为 3.

计算 Jordan 分解必须求助非正交变换. 关于这类约化, 我们建议参考 Golub and Wilkinson(1976) 或 Kågström and Ruhe(1980a,1980b).

上面 SVD 的计算充分显示了每步都要困难地决定秩以及最后计算出来块结构严重依赖于那些决定. 侥幸的是, 在实际应用中几乎总是能用稳定的 Schur 分解来代替 Jordan 分解.

## 习 题

**7.6.1** 试给出一算法用来解决一个实的 $n \times n$ 上拟三角形方程组 $\boldsymbol{Tx} = \boldsymbol{b}$.

**7.6.2** 设 $U^{-1}AU = \text{diag}(\alpha_1, \cdots, \alpha_m)$ 且 $V^{-1}BV = \text{diag}(\beta_1, \cdots, \beta_n)$, 证明：若 $\phi(X) = AX + XB$, 则 $\lambda(\phi) = \{\alpha_i + \beta_j : i = 1 : m, j = 1 : n\}$. 请问, 相应的特征向量是什么? 如何用这些分解来求解 $AX + XB = C$?

**7.6.3** 证明若 $Y = \begin{bmatrix} I & Z \\ 0 & I \end{bmatrix}$ 则 $\kappa_2(Y) = [2 + \sigma^2 + \sqrt{4\sigma^2 + \sigma^4}]/2$, 其中 $\sigma = \|Z\|_2$.

**7.6.4** 导出 (7.6.5) 式.

**7.6.5** 设 $T \in \mathbb{R}^{n\times n}$ 是分块上三角形矩阵, 其划分如下：

$$T = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & T_{33} \end{bmatrix}, \quad T \in \mathbb{R}^{n\times n}.$$

假定对角块 $T_{22}$ 是 $2 \times 2$ 的, 其特征值是复数且与 $\lambda(T_{11})$ 和 $\lambda(T_{33})$ 不相交, 试给出计算与 $T_{22}$ 的特征值所对应的 2 维实的不变子空间的算法.

**7.6.6** 假设 $H \in \mathbb{R}^{n\times n}$ 是具有复特征值 $\lambda + \mathrm{i}\mu$ 的上 Hessenberg 矩阵, 怎样用逆迭代来计算 $x, y \in \mathbb{R}^n$ 使得 $H(x + \mathrm{i}y) = (\lambda + \mathrm{i}\mu)(x + \mathrm{i}y)$ ? 提示：比较等式中实部和复部, 就得到一个 $2n \times 2n$ 实方程组.

**7.6.7** (a) 证明：如果 $\mu_0 \in \mathbb{C}$ 有非零实部, 则迭代

$$\mu_{k+1} = \frac{1}{2}\left(\mu_k + \frac{1}{\mu_k}\right)$$

在 $\text{Re}(\mu_0) > 0$ 时收敛到 1, 在 $\text{Re}(\mu_0) < 0$ 时收敛到 $-1$.

(b) 设 $A \in \mathbb{C}^{n\times n}$ 是可对角化, 且

$$A = X \begin{bmatrix} D_+ & 0 \\ 0 & D_- \end{bmatrix} X^{-1},$$

其中 $D_+ \in \mathbb{C}^{p\times p}, D_- \in \mathbb{C}^{(n-p)\times(n-p)}$ 分别是特征值位于右半开平面和左半开平面的对角矩阵. 证明迭代

$$A_{k+1} = \frac{1}{2}(A_k + A_k^{-1}), \quad A_0 = A$$

收敛到 $\text{sign}(A) \equiv X \begin{bmatrix} I_p & 0 \\ 0 & -I_{n-p} \end{bmatrix} X^{-1}$.

(c) 设 $M = \begin{array}{cc} \begin{bmatrix} M_{11} & M_{12} \\ 0 & M_{22} \end{bmatrix} & \begin{matrix} p \\ n-p \end{matrix} \\ \begin{matrix} p & n-p \end{matrix} & \end{array}$ 具有性质：$\lambda(M_{11})$ 在右半开平面且 $\lambda(M_{22})$ 在左半开平面. 证明：

$$\text{sign}(M) = \begin{bmatrix} I_p & Z \\ 0 & -I_{n-p} \end{bmatrix},$$

且 $-Z/2$ 是 $M_{11}X - XM_{22} = -M_{12}$ 的解. 这样,

$$U = \begin{bmatrix} I_p & -Z/2 \\ 0 & I_{n-p} \end{bmatrix} \Rightarrow U^{-1}MU = \begin{bmatrix} M_{11} & 0 \\ 0 & M_{22} \end{bmatrix}.$$

## 本节注释与参考文献

本书所讨论的许多内容可以在下面综述性文章中找到：

G. H. Golub and J. H. Wilkinson(1976). "Ill-Conditioned Eigensystems and the Computation of the Jordan Canonical Form," *SIAM Review* 18, 578–619.

专门分析计算特征向量的逆迭代法的论文包括：

J. Varah(1968). "The Calculation of the Eigenvectors of a General Complex Matrix by Inverse Iteration," *Math. Comp.* 22, 785–791.

J. Varah(1968). "Rigorous Machine Bounds for the Eigensystem of a General Complex Matrix," *Math. Comp.* 22, 793–801.

J. Varah(1970). "Computing Invariant Subspaces of a General Matrix When the Eigensystem is Poorly Determined," *Math. Comp.* 24, 137–149.

G. Peters and J. H. Wilkinson(1979). "Inverse Iteration,Ill-Conditioned Equations, and Newton's Method," *SIAM Review* 21, 339–360.

Eispack 逆迭代的子程序的 Algol 版本见：

G. Peters and J. H. Wilkinson(1971). "The Calculation of Specified Eigenvectors by Inverse Iteration," in Wilkinson and Reinsch(1971, pp. 418–439).

下面论文主要讨论了特征值排序问题：

A. Ruhe(1970). "An Algorithm for Numerical Determination of the Structure of a General Matrix," *BIT* 10, 196–216.

G. W. Stewart(1976). "Algorithm 406: HQR3 and EXCHNG: Fortran Subroutines for Galculating and Ordering the Eigenvalues of a Real Upper Hessenberg Matrix," *ACM Trans.Math. Soft.* 2, 275–280.

J. J. Dongarra, S. Hammarling, and J. H. Wilkinson(1992). "Numerical Considerations in Computing Invariant Subspaces," *SIAM J.Matrix Anal. Appl.* 13, 145–161.

Z. Bai and J. W. Demmel(1993). "On Swapping Diagonal Blocks in Real Schur Form," *Lin. Alg. and Its Applic.* 186, 73–95.

计算分块对角化 Jordan 型的 Fortran 程序见：

C. Bavely and G. W. Stewart(1979). "An Algorithm for Computing Reducing Subspaces by Block Diagonalization," *SIAM J. Num. Anal.* 16, 359–367.

B. Kågström and A. Ruhe(1980a). "An Algorithm for Numerical Computation of the Jordan Normal Form of a Complex Matrix," *ACM Trans. Math. Soft.* 6, 398–419.

B. Kågström and A. Ruhe(1980b). "Algorithm 560 JNF: An Algorithm for Numerical Computation of the Jordan Normal Form of a Complex Matrix," *ACM Trans. Math. Soft.* 6, 437–443.

J. W. Demmel(1983). "A Numerical Analyst's Jordan Canonical Form," Ph. D. Thesis, Berkeley.

有关在计算特征值和 (或) 特征向量中误差估计的文章有:

S. P. Chan and B. N. Parlett(1977). "Algorithm 517: A Program for Computing the Condition Numbers of Matrix Eigenvalues Without Computing Eigenvectors," *ACM Trans. Math. Soft.* 3, 186–203.

H. J. Symm and J. H. Wilkinson(1980). "Realistic Error Bounds for a Simple Eigenvalue and Its Associated Eigenvector," *Numer. Math.* 35, 113–126.

C. Van Loan(1987). "On Estimating the Condition of Eigenvalues and Eigenvectors," *Lin. Alg. and Its Applic.* 88/89, 715–732.

Z. Bai, J. Demmel, and A. McKenney(1993). "On Computing Condition Numbers for the Nonsymmetric Eigenproblem," *ACM Trans. Math.Soft.* 19, 202–223.

如我们所见, sep(.,.) 函数在评估计算不变子空间中非常重要. 此量的性质和相关的 Sylvester 方程讨论于:

J. Varah(1979). "On the Separation of Two Matrices," *SIAM J. Num. Anal.* 16, 212–222.

R. Byers(1984). "A Linpack-Style Condition Estimator for the Equation $AX - XB^{\mathrm{T}} = C$," *IEEE Trans. Auto. Cont.* AC-29, 926–928.

K. Datta(1988). "The Matrix Equation $XA - BX = R$ and Its Applications," *Lin. Alg. and Its Appl.* 109,91–105.

N. J. Higham(1993). "Perturbation Theory and Backward Error for AX-XB=C," *BIT* 33, 124–136.

J. Gardiner, M. R.Wette, A. J. Laub, J. J. Amato, and C. B. Moler(1992). "Algorithm 705: A FORTRAN-77 Software Package for Solving the Sylvester Matrix Equation $AXB^{\mathrm{T}} + CXD^{\mathrm{T}} = E$," *ACM Trans. Math. Soft.* 18, 232–238.

针对 Sylveter 方程已经提出了一些算法, 而在下面文章中描述的算法较可靠, 因为它们依赖正交变换:

R. H. Bartels and G. W. Stewart(1972). "Solution of the Equation$AX + XB = C$," *Comm. ACM* 15, 820–826.

G. H. Golub, S. Nash, and C. Van Loan(1979). "A Hessenberg-Schur Method for the Matrix Problem $AX + XB = C$," *IEEE Trans. Auto. Cont.* AC-24, 909–913.

一个带约束的 Sylvester 方程见于:

J. B. Barlow, M. M. Monahemi, and D. P. O'Leary(1992). "Constrained Matrix Sylvester Equations," *SIAM J Matrix Anal. Appl.* 13, 1–9.

Lyapunov 问题 $\boldsymbol{FX} + \boldsymbol{XF}^{\mathrm{T}} = -\boldsymbol{C}$, 其中 $\boldsymbol{C}$ 非负定, 在控制理论中扮演着非常重要的角色. 见:

S. Barnett and C. Storey(1968). "Some Applications of the Lyapunov Matrix Equation, " *J.Inst. Math. Applic.* 4, 33–42.

G. Hewer and C. Kenney(1988). "The Sensitivity of the Stable Lyapunov Equation," *SIAMJ. Control Optim* 26, 321–344.

A. R. Ghavimi and A. J. Laub(1995). "Residual Bounds for Discrete-Time Lyapunov Equations," *IEEE Trans.Auto. Cont.* 40, 1244–1249.

一些作者已经考虑到了推广的 Sylvester 方程, 即 $\sum \boldsymbol{F}_i \boldsymbol{X} \boldsymbol{G}_i = \boldsymbol{C}$, 包括:

P. Lancaster(1970). "Explicit Solution of Linear Matrix Equations," *SIAM Review* 12, 544–566.

H. Wimmer and A. D. Ziebur(1972). "Solving the Matrix Equations$\sum f_p(A)g_p(A) = C$," *SIAM Review* 14, 318–323.

W. J. Vetter(1975). "Vector Structures and Solutions of Linear Matrix Equations," *Lin. Alg. and Its Applic.* 10, 181–188.

对计算出来的特征值、特征向量和不变子空间进行改进的一些思想可在下面文章中找到:

J. J. Dongarra, C. B. Moler, and J. H. Wilkinson(1983). "Improving the Accuracy of Computed Eigenvalues and Eigenvectors," *SIAM J. Numer. Anal.* 20, 23–46.

J. W. Demmel(1987). "Three Methods for Refining Estimates of Invariant Subspaces," *Computing* 38, 43–57.

Hessenberg 和 QR 迭代技巧计算快, 但不适合并行计算. 因为这点, 急须找到求特征问题的一种较完全的新算法. 这里的一些文章集中谈到矩阵符号函数和有很高表现潜能的相关思想:

C. S. Kenney and A. J. Laub(1991). "Rational Iterative Methods for the Matrix Sign Function," *SIAM J. Matrix Anal.Appl.* 12, 273–291.

C. S. Kenney, A. J. Laub, and P. M. Papadopouos(1992). "Matrix Sign Algorithms for Riccati Equations," *IMA J. of Math. Control Inform.* 9, 331–344.

C. S. Kenney and A. J. Laub(1992). "On Scaling Newton's Method for Polar Decomposition and the Matrix Sign Function," *SIAM J. Matrix Anal. Appl.* 13, 688–706.

N. J. Higham(1994). "The Matrix Sign Decomposition and Its Relation to the Polar Decomposition," *Lin. Alg. and Its Applic* 212/213, 3–20.

L. Adams and P. Arbenz(1994). "Towards a Divide and Conquer Algorithm for the Real Nonsymmetric Eigenvalue Problem," *SIAM J. Matrix Anal. Appl* 15, 1333–1353.

## 7.7 $Ax = \lambda Bx$ 的 QZ 方法

令 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 是两个 $n \times n$ 矩阵. 所有形如 $\boldsymbol{A} - \lambda \boldsymbol{B}$ , $(\lambda \in \mathbb{C})$ 的矩阵的集合称为**束**. 束的特征值是集 $\lambda(A, B)$ 的元素, 定义为:

$$\lambda(\boldsymbol{A},\boldsymbol{B})=\{z\in\mathbb{C}:\det(\boldsymbol{A}-z\boldsymbol{B})=0\}.$$

如果 $\lambda\in\lambda(\boldsymbol{A},\boldsymbol{B})$, 且

$$\boldsymbol{Ax}=\lambda\boldsymbol{Bx},\quad \boldsymbol{x}\neq\boldsymbol{0},\tag{7.7.1}$$

则称 $\boldsymbol{x}$ 为 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 的特征向量.

本节简要地综述广义特征问题 (7.7.1) 的某些数学性质, 并为解决这个问题提供一个稳定的方法. 在 8.7.2 节中讨论 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是对称矩阵且后者正定的重要情形.

### 7.7.1 基本知识

观察广义特征值问题的第一件事是: 当且仅当 $\text{rank}(\boldsymbol{B})=n$ 时它存在 $n$ 个特征值. 如果 $\boldsymbol{B}$ 为秩亏损阵, 则 $\lambda(\boldsymbol{A},\boldsymbol{B})$ 可能是有限集、空集或无限集.

$$\boldsymbol{A}=\begin{bmatrix}1&2\\0&3\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}1&0\\0&0\end{bmatrix}\Rightarrow\lambda(\boldsymbol{A},\boldsymbol{B})=\{1\},$$

$$\boldsymbol{A}=\begin{bmatrix}1&2\\0&3\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}0&1\\0&0\end{bmatrix}\Rightarrow\lambda(\boldsymbol{A},\boldsymbol{B})=\varnothing,$$

$$\boldsymbol{A}=\begin{bmatrix}1&2\\0&0\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}1&0\\0&0\end{bmatrix}\Rightarrow\lambda(\boldsymbol{A},\boldsymbol{B})=\mathbb{C}.$$

注意到, 若 $0\neq\lambda\in\lambda(\boldsymbol{A},\boldsymbol{B})$, 则 $(1/\lambda)\in\lambda(\boldsymbol{B},\boldsymbol{A})$, 此外, 如果 $\boldsymbol{B}$ 为非奇异矩阵, 则 $\lambda(\boldsymbol{A},\boldsymbol{B})=\lambda(\boldsymbol{B}^{-1}\boldsymbol{A},\boldsymbol{I})=\lambda(\boldsymbol{B}^{-1}\boldsymbol{A})$.

上面的观察为 $\boldsymbol{B}$ 非奇异时, 提供了一个解 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 问题的方法:

- 解 $\boldsymbol{BC}=\boldsymbol{A}$ 求出 $\boldsymbol{C}$, 比方说可用列选主的高斯消去法;
- 用 QR 算法计算 $\boldsymbol{C}$ 的特征值.

注意, $\boldsymbol{C}$ 将受到数量级为 $\mathbf{u}\|\boldsymbol{A}\|_2\|\boldsymbol{B}^{-1}\|_2$ 的舍入误差之影响. 如果 $\boldsymbol{B}$ 是病态的, 则这就排除了精确地计算任何一个广义特征值的可能性 —— 包括被认为是良态的特征值.

**例 7.7.1** 如果

$$\boldsymbol{A}=\begin{bmatrix}1.746&0.940\\1.246&1.898\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}0.780&0.563\\0.913&0.659\end{bmatrix},$$

则 $\lambda(\boldsymbol{A},\boldsymbol{B})=\{2,1.07\times10^6\}$. 用 7 位浮点运算, 发现 $\lambda(fl(\boldsymbol{AB}^{-1}))=\{1.562\,539,1.01\times10^6\}$, 小特征值精度很差的原因是 $\kappa_2(\boldsymbol{B})\approx2\times10^6$. 另一方面, 我们发现

$$\lambda(\boldsymbol{I},fl(\boldsymbol{A}^{-1}\boldsymbol{B}))\approx\{2.000\,001,1.06\times10^6\}.$$

由于 $\kappa_2(\boldsymbol{A})\approx4$, 小特征值的精度被改善了.

这个例子提醒我们去寻找其他解 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 问题的方法. 一个想法是选取良态矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 使得矩阵

$$A_1 = Q^{-1}AZ, \quad B_1 = Q^{-1}BZ \tag{7.7.2}$$

都是标准型. 注意到, 从

$$Ax = \lambda Bx \Longleftrightarrow A_1 y = \lambda B_1 y, \quad x = Zy,$$

可知 $\lambda(A,B)=\lambda(A_1,B_1)$. 如果 (7.7.2) 成立且 $Q$ 和 $Z$ 非奇异, 我们就说两个束 $A-\lambda B$ 和 $A_1-\lambda B_1$等价.

### 7.7.2 广义 Schur 分解

与标准特征问题 $A-\lambda I$ 一样, 在标准型当中要有选择. 与 Jordan 型类似的是 Kronecher 的分解, 在此分解中 $A_1$ 和 $B_1$ 都是块对角的. 这些块相似于 Jordan 块. Kronecher 标准型面临和 Jordan 型同样的数值困难. 然而, 这个分解的确能揭露束 $A-\lambda B$ 的数学性质. 详见 Wilkinson(1978) 和 Demmel and Kågström(1987).

从数值观点来看, 更吸引人的是由 Moler and Stewert(1973) 描述的下列分解.

**定理 7.7.1(广义 Schur 分解)** *如果 $A,B\in\mathbb{C}^{n\times n}$, 则存在酉矩阵 $Q$ 和 $Z$ 使得 $Q^{\mathrm{H}}AZ=T$和$Q^{\mathrm{H}}BZ=S$ 是上三角形矩阵. 若对某个 $k$, $t_{kk}$ 和 $s_{kk}$ 都为零, 则 $\lambda(A,B)=\mathbb{C}$, 否则*

$$\lambda(A,B)=\{t_{ii}/s_{ii}: s_{ii}\neq 0\}.$$

**证明** 令 $\{B_k\}$ 是收敛于 $B$ 的一列非奇异矩阵. 对每个 $k$, 令 $Q_k^{\mathrm{H}}(AB_k^{-1})Q_k=R_k$ 为 $AB_k^{-1}$ 的 Schur 分解. 令 $Z_k$ 是使 $Z_k^{\mathrm{H}}(B_k^{-1}Q_k)\equiv S_k^{-1}$ 为上三角形矩阵的酉矩阵. 由此可见, $Q_k^{\mathrm{H}}AZ_k=R_kS_k$ 和 $Q_k^{\mathrm{H}}B_kZ_k$ 都是上三角形矩阵.

运用 Bolzano-Weierstrass 定理, 我们知道有界序列 $\{(Q_k,Z_k)\}$ 有收敛子列, $\lim(Q_{k_i},Z_{k_i})=(Q,Z)$. 易证 $Q$ 和 $Z$ 是酉矩阵且 $Q^{\mathrm{H}}AZ$ 和 $Q^{\mathrm{H}}BZ$ 都是上三角形矩阵. 从等式

$$\det(A-\lambda B)=\det(QZ^{\mathrm{H}})\prod_{i=1}^{n}(t_{ii}-\lambda s_{ii})$$

即得到关于 $\lambda(A,B)$ 的断言. □

若 $A,B$ 是实矩阵, 则对应实 Schur 分解 (定理 7.4.1) 的下列分解是重要的:

**定理 7.7.2(推广的实 Schur 分解)** *如果 $A$ 和 $B$ 是 $\mathbb{R}^{n\times n}$ 中的矩阵, 则存在正交矩阵 $Q$ 和 $Z$ 使得 $Q^{\mathrm{T}}AZ$ 是拟上三角形矩阵且 $Q^{\mathrm{T}}BZ$ 是上三角形矩阵.*

**证明** 参看 Stewart(1972). □

在本节其余部分, 我们考虑此分解的计算和它的数学内涵.

### 7.7.3 敏感度

广义 Schur 分解阐明了 $A-\lambda B$ 问题的特征值敏感性. 很明显, 如果 $s_{ii}$ 很小, $A$ 和 $B$ 的微小变化能够导致特征值 $\lambda_i=t_{ii}/s_{ii}$ 较大的变化. 然而, 如 Stewart(1978) 所说, 认为这样的特征值是 “病态的” 是不恰当的. 理由在于其倒数 $\mu_i=s_{ii}/t_{ii}$ 可能是束 $\mu A-B$ 的性质很好的特征值. 在 Stewert 的分析中, $A$ 和 $B$ 被平等看待且

特征值看成有序对 $(t_{ii}, s_{ii})$ 而不是商. 按此观点, 很适合用弦度来度量特征值的扰动, 弦度 chord$(a,b)$ 定义为

$$\text{chord}(a,b) = \frac{|a-b|}{\sqrt{1+a^2}\sqrt{1+b^2}}.$$

Stewart 证明了若 $\lambda$ 是 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 的单特征值且 $\lambda_\varepsilon$ 是相应扰动束 $\tilde{\boldsymbol{A}} - \lambda\tilde{\boldsymbol{B}}$ 的特征值, 这里 $\|\boldsymbol{A}-\tilde{\boldsymbol{A}}\|_2 \approx \|\boldsymbol{B}-\tilde{\boldsymbol{B}}\|_2 \approx \varepsilon$, 则

$$\text{chord}(\lambda, \lambda_\varepsilon) \leqslant \frac{\varepsilon}{(\boldsymbol{y}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{x})^2 + (\boldsymbol{y}^{\mathrm{H}}\boldsymbol{B}\boldsymbol{x})^2} + O(\varepsilon^2),$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 都是 2 范数单位向量, 满足 $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{B}\boldsymbol{x}$ 和 $\boldsymbol{y}^{\mathrm{H}} = \lambda\boldsymbol{y}^{\mathrm{H}}\boldsymbol{B}$. 注意到, 上界中的分母关于 $\boldsymbol{A},\boldsymbol{B}$ 是对称的. "真正" 的病态特征值是那些使得该分母很小的特征值.

对某个 $k, t_{kk} = s_{kk} = 0$ 的极端情形已被 Wiklinson(1979) 所研究. 他有一个有趣的观察, 当此情形发生时, 其他的商 $t_{ii}/s_{ii}$ 可假定为任何值.

### 7.7.4 Hessenberg 三角型

计算矩阵对 $(\boldsymbol{A},\boldsymbol{B})$ 的广义 Schur 分解的第一步是通过正交变换化 $\boldsymbol{A}$ 为上 Hessenberg 型, 化 $\boldsymbol{B}$ 为上三角型. 我们首先确定一个正交矩阵 $\boldsymbol{U}$, 使得 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{B}$ 是上三角形矩阵. 当然, 为保持特征值不变, 我们也必须用同一正交矩阵作用于 $\boldsymbol{A}$. 让我们看看 $n=5$ 时的情况.

$$\boldsymbol{A} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix}, \quad \boldsymbol{B} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{B} = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

接着, 我们在保持 $\boldsymbol{B}$ 为上三角型时, 化 $\boldsymbol{A}$ 为上 Hessenberg 型, 首先, 用 Givens 旋转变换 $\boldsymbol{Q}_{45}$ 消 $a_{51}$ 为零:

$$\boldsymbol{A} = \boldsymbol{Q}_{45}^{\mathrm{T}}\boldsymbol{A} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{bmatrix}, \quad \boldsymbol{B} = \boldsymbol{Q}_{45}^{\mathrm{T}}\boldsymbol{B} = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

在 $\boldsymbol{B}$ 中 (5,4) 位置产生的非零元素能通过右乘适当的 Givens 旋转 $\boldsymbol{Z}_{45}$ 消为零:

$$\boldsymbol{A}=\boldsymbol{A}\boldsymbol{Z}_{45}=\begin{bmatrix}\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\end{bmatrix},\quad \boldsymbol{B}=\boldsymbol{B}\boldsymbol{Z}_{45}=\begin{bmatrix}\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&0&\times&\times&\times\\0&0&0&\times&\times\\0&0&0&0&\times\end{bmatrix}.$$

类似地, 可将 $\boldsymbol{A}$ 的 (4.1) 元和 (3,1) 元消为零:

$$\boldsymbol{A}=\boldsymbol{Q}_{34}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\end{bmatrix},\quad \boldsymbol{B}=\boldsymbol{Q}_{34}^{\mathrm{T}}\boldsymbol{B}=\begin{bmatrix}\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&0&\times&\times&\times\\0&0&\times&\times&\times\\0&0&0&0&\times\end{bmatrix},$$

$$\boldsymbol{A}=\boldsymbol{A}\boldsymbol{Z}_{34}=\begin{bmatrix}\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\end{bmatrix},\quad \boldsymbol{B}=\boldsymbol{B}\boldsymbol{Z}_{34}=\begin{bmatrix}\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&0&\times&\times&\times\\0&0&0&\times&\times\\0&0&0&0&\times\end{bmatrix},$$

$$\boldsymbol{A}=\boldsymbol{Q}_{23}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\end{bmatrix},\quad \boldsymbol{B}=\boldsymbol{Q}_{23}^{\mathrm{T}}\boldsymbol{B}=\begin{bmatrix}\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&0&0&\times&\times\\0&0&0&0&\times\end{bmatrix},$$

$$\boldsymbol{A}=\boldsymbol{A}\boldsymbol{Z}_{23}=\begin{bmatrix}\times&\times&\times&\times&\times\\\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&\times&\times&\times&\times\end{bmatrix},\quad \boldsymbol{B}=\boldsymbol{B}\boldsymbol{Z}_{23}=\begin{bmatrix}\times&\times&\times&\times&\times\\0&\times&\times&\times&\times\\0&0&\times&\times&\times\\0&0&0&\times&\times\\0&0&0&0&\times\end{bmatrix}.$$

$\boldsymbol{A}$ 的第一列已是上 Hessenberg 型. 将 $a_{52}$, $a_{42}$ 和 $a_{53}$ 消为零, 此约化则完成. 很明显, 要将每个 $a_{ij}$ 化为零, 需要两个正交变换 —— 一个用于化零, 另一个用于恢复 $\boldsymbol{B}$ 的三角形式. 用 Givens 旋转或 $2\times 2$ 修正的 Householder 变换均可. 总之, 我们有下列算法.

**算法 7.7.1 (Hessenberg 三角形约化)** 令 $\boldsymbol{A}$ 和 $\boldsymbol{B}\in\mathbb{R}^{n\times n}$, 本算法计算上 Hessenberg 矩阵 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Z}$ 和上三角形矩阵 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}$ 并分别覆盖矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$, 其中 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 是正交矩阵.

用算法 5.2.1, 计算 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}=\boldsymbol{R}$ 并覆盖 $\boldsymbol{B}$, 其中 $\boldsymbol{Q}$ 是正交矩阵且 $\boldsymbol{R}$ 为上三角形矩阵.

$\boldsymbol{A}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}$

**for** $j=1:n-2$

  **for** $i=n:-1:j+2$

$[c,s]=\textbf{givens}(A(i-1,j)A(i,j))$

$$A(i-1:i,j:n)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} A(i-1:i,j:n)$$

$$B(i-1:i,i-1:n)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} B(i-1:i,i-1:n)$$

$[c,s]=\textbf{givens}(-B(i,i),B(i,i-1))$

$$B(1:i,i-1:i)=B(1:i,i-1:i)\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

$$A(1:n,i-1:i)=A(1:n,i-1:i)\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

  **end**

**end**

此算法需 $8n^3$ 个 flop. 要把 $\boldsymbol{Q},\boldsymbol{Z}$ 累积算出来还分别需要 $4n^3$ 个 flop 和 $3n^3$ 个 flop.

化 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 为 Hessenberg 三角型可用作一种广义的 QR 迭代, 即下节要介绍的称为 QZ 迭代的 "前期" 分解.

**例 7.7.2** 设

$$\boldsymbol{A}=\begin{bmatrix} 10 & 1 & 2 \\ 1 & 2 & -1 \\ 1 & 1 & 2 \end{bmatrix}, \quad \boldsymbol{B}=\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

且正交矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 定义为

$$\boldsymbol{Q}=\begin{bmatrix} -0.1231 & -0.9917 & 0.0378 \\ -0.4924 & 0.0279 & -0.8699 \\ -0.8616 & 0.1257 & 0.4917 \end{bmatrix}$$

$$\boldsymbol{Z}=\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & -0.8944 & -0.4472 \\ 0.0000 & 0.4472 & -0.8944 \end{bmatrix},$$

则 $\boldsymbol{A}_1=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Z}$ 且 $\boldsymbol{B}_1=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}$ 为

$$\boldsymbol{A}_1=\begin{bmatrix} -2.5849 & 1.5413 & 2.4221 \\ -9.7631 & 0.0874 & 1.9239 \\ 0.0000 & 2.7233 & -0.7612 \end{bmatrix},$$

$$B_1 = \begin{bmatrix} -8.1240 & 3.6332 & 14.2024 \\ 0.0000 & 0.0000 & 1.8739 \\ 0.0000 & 0.0000 & 0.7612 \end{bmatrix}.$$

### 7.7.5 降阶

不失一般性, 在描述 QZ 迭代时, 我们可以假定 $A$ 是不可约上 Hessenberg 矩阵, $B$ 为非奇异上三角形矩阵. 第一个要求显然合理, 因为若 $a_{k+1,k} = 0$, 则

$$A - \lambda B = \begin{bmatrix} A_{11} - \lambda B_{11} & A_{12} - \lambda B_{12} \\ \mathbf{0} & A_{22} - \lambda B_{22} \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix}$$
$$\qquad\qquad k \qquad\qquad n-k$$

从而我们只需求解两个较小的问题 $A_{11} - \lambda B_{11}$ 和 $A_{22} - \lambda B_{22}$. 另一方面, 若对某个 $k$ 有 $b_{kk} = 0$, 则可把 $A$ 的 $(n, n-1)$ 位置的元素化为零, 然后降阶. 下面举列说明, 设 $n = 5, k = 3$:

$$A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad B = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

用 Givens 旋转可以把 $B$ 对角线上的零 "往下推" 到 (5,5) 位置.

$$A = Q_{34}^{\mathrm{T}} A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad B = Q_{34}^{\mathrm{T}} B = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix},$$

$$A = AZ_{23} = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad B = BZ_{23} = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix},$$

$$A = Q_{45}^{\mathrm{T}} A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}, \quad B = Q_{45}^{\mathrm{T}} B = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$
\boldsymbol{A}=\boldsymbol{A}\boldsymbol{Z}_{34}=\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad \boldsymbol{B}=\boldsymbol{B}\boldsymbol{Z}_{34}^{\mathrm{T}}=\begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},
$$

$$
\boldsymbol{A}=\boldsymbol{A}\boldsymbol{Z}_{45}=\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix}, \quad \boldsymbol{B}=\boldsymbol{B}\boldsymbol{Z}_{45}=\begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.
$$

这个零追逐技巧完全是通用的, 不论零出现在 $\boldsymbol{B}$ 的对角线什么位置, 均可将 $a_{n,n-1}$ 化为零.

### 7.7.6 QZ 步

现在, 我们来描述 QZ 步. 其基本思想是把 $\boldsymbol{A},\boldsymbol{B}$ 作如下的变换:

$$
(\bar{\boldsymbol{A}}-\lambda\bar{\boldsymbol{B}})=\bar{\boldsymbol{Q}}^{\mathrm{T}}(\boldsymbol{A}-\lambda\boldsymbol{B})\bar{\boldsymbol{Z}},
$$

其中 $\bar{\boldsymbol{A}}$ 是上 Hessenberg 矩阵, $\bar{\boldsymbol{B}}$ 是上三角形矩阵, $\bar{\boldsymbol{Q}}$ 和 $\bar{\boldsymbol{Z}}$ 均为正交矩阵, 且 $\bar{\boldsymbol{A}}\,\bar{\boldsymbol{B}}^{-1}$“本质上” 和将 Francis QR 步 (算法 7.5.2) 应用于 $\boldsymbol{A}\boldsymbol{B}^{-1}$ 所产生的矩阵是同一个矩阵. 我们用若干巧妙的零追逐技术以及求助于隐式 Q 定理可以做到这一点.

令 $\boldsymbol{M}=\boldsymbol{A}\boldsymbol{B}^{-1}$(上 Hessenberg 型) 且令 $v$ 是矩阵 $(\boldsymbol{M}-a\boldsymbol{I})(\boldsymbol{M}-b\boldsymbol{I})$ 的第一列, 其中 $a$ 和 $b$ 是 $\boldsymbol{M}$ 的右下角 $2\times 2$ 子矩阵的特征值. 注意, 可用 $O(1)$ 个 flop 计算出 $v$. 若 $\boldsymbol{P}_0$ 是使 $\boldsymbol{P}_0 v$ 为 $e_1$ 之倍数的 Householder 矩阵, 则

$$
\boldsymbol{A}=\boldsymbol{P}_0\boldsymbol{A}=\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad \boldsymbol{B}=\boldsymbol{P}_0\boldsymbol{B}=\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}.
$$

现在的思路是通过往下赶对角线之下多余的非零元来把这两个矩阵分别恢复到 Hessenberg 三角型.

为此, 先确定一对 Householder 矩阵 $\boldsymbol{Z}_1$ 和 $\boldsymbol{Z}_2$ 将 $b_{31}$, $b_{32}$ 和 $b_{21}$ 化为零:

$$A = AZ_1Z_2 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix},$$

$$B = BZ_1Z_2 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

然后, 用 Householder 矩阵 $P_1$ 将 $a_{31}$ 和 $a_{41}$ 化为零:

$$A = P_1A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad B = P_1B = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

注意到, 至此, 不想要的非零元已从它们的原来位置向右下方移动. 这描述 QZ 算法的一个典型步骤. 注意到 $Q = Q_0Q_1\cdots Q_{n-1}$ 和 $Q_0$ 有相同的第一列. 根据初始 Householder 矩阵的确定方式, 我们可以利用隐式 Q 定理, 断言 $AB^{-1} = Q^{\mathrm{T}}(AB^{-1})Q$ 的确 "本质上" 与直接将 Francis 迭代用于 $M = AB^{-1}$ 所得到的矩阵是同一矩阵. 总之我们有下列算法.

**算法 7.7.2(QZ 步)** 给定不可约上 Hessenberg 矩阵 $A \in \mathbb{R}^{n\times n}$ 和非奇异上三角形矩阵 $B \in \mathbb{R}^{n\times n}$, 本算法用 Hessenberg 矩阵 $Q^{\mathrm{T}}AZ$ 和上三角形矩阵 $Q^{\mathrm{T}}BZ$ 分别覆盖 $A$ 和 $B$, 其中 $Q$ 和 $Z$ 为正交矩阵, 而且 $Q$ 与应用于 $AB^{-1}$ 之算法 7.5.1 的正交相似变换矩阵有相同的第 1 列.

令 $M = AB^{-1}$, 计算 $(M - aI)(M - bI)e_1 = (x, y, z, 0, \cdots, 0)^{\mathrm{T}}$,

  其中 $a$ 和 $b$ 为 $M$ 的右下角 $2\times 2$ 的主子矩阵的特征值

**for** $k = 1 : n-2$

  找 Householder 矩阵 $Q_k$ 使得 $Q_k[x\ y\ z]^{\mathrm{T}} = [*\ 0\ 0]^{\mathrm{T}}$.

  $A = \mathrm{diag}(I_{k-1}, Q_k, I_{n-k-2})A$

  $B = \mathrm{diag}(I_{k-1}, Q_k, I_{n-k-2})B$

  找 Householder 矩阵 $Z_{k1}$ 使得

$$[b_{k+2,k} \quad b_{k+2,k+1} \quad b_{k+2,k+2}]Z_{k_1} = [0\ 0\ *].$$

$A = A\ \mathrm{diag}(I_{k-1}, Z_{k1}, I_{n-k-2})$

$B = B\ \mathrm{diag}(I_{k-1}, Z_{k1}, I_{n-k-2})$

找 Householder 矩阵 $Z_{k2}$ 使

$$[b_{k+1,k} \quad b_{k+1,k+1}]Z_{k2} = [0 \quad *]$$

$A = A\ \mathrm{diag}(I_{k-1}, Z_{k2}, I_{n-k-1})$

$B = B\ \mathrm{diag}(I_{k-1}, Z_{k2}, I_{n-k-1})$

$x = a_{k+1,k}; y = a_{k+1,k}$

**if** $k < n-2$

$z = a_{k+3,k}$

**end**

**end**

找 Householder 矩阵 $\boldsymbol{Q}_{n-1}$ 使得 $\boldsymbol{Q}_{n-1}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix}$

$A = \mathrm{diag}(I_{n-2}, \boldsymbol{Q}_{n-1})A$

$B = \mathrm{diag}(I_{n-2}, \boldsymbol{Q}_{n-1})B$

找 Householder 矩阵 $Z_{n-1}$ 使得

$$[b_{n,n-1} \quad b_{nn}]Z_{n-1} = [0 \quad *]$$

$A = A\mathrm{diag}(I_{n-2}, Z_{n-1})$

$B = B\mathrm{diag}(I_{n-2}, Z_{n-1})$

这个算法需 $22n^2$ 个 flop. 累积 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 分别需额外的 $8n^2$ 个 flop 和 $13n^2$ 个 flop.

### 7.7.7 完整 QZ 过程

把一系列 QZ 步应用到 Hessenberg 三角形束 $\boldsymbol{A} - \lambda\boldsymbol{B}$, 就能将 $\boldsymbol{A}$ 化为拟三角型. 在运算中, 有必要监视 $\boldsymbol{A}$ 的次对角元和 $\boldsymbol{B}$ 的对角元, 若有可能就进行解耦. 完整的过程 (由 Moler and Stewart(1973) 提出) 如下.

**算法 7.7.3** 给定 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times n}$, 本算法计算正交矩阵 $\boldsymbol{Q}$ 和 $\boldsymbol{Z}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{AZ} = \boldsymbol{T}$ 为上拟三角形矩阵且 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{BZ} = \boldsymbol{S}$ 是上三角形矩阵. $\boldsymbol{T}$ 覆盖 $\boldsymbol{A}$, $\boldsymbol{S}$ 覆盖 $\boldsymbol{B}$.

用算法 7.7.1 计算 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{AZ}$(上 Hessenberg 矩阵) 覆盖 $\boldsymbol{A}$

和 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{BZ}$ 覆盖 $\boldsymbol{B}$(上三角形矩阵)

**until** $q = n$

令所有满足

$$|a_{i,i-1}| \leqslant \varepsilon(|a_{i-1,i-1}| + |a_{ii}|)$$

的次对角元素为零. 找到最大非负值 $q$ 和最小非负值 $p$ 使得如果

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$

$$\begin{matrix} p & n-p-q & q \end{matrix}$$

则 $A_{33}$ 是上拟三角形矩阵且 $A_{22}$ 不可约化为 Hessenberg 矩阵.

把 $B$ 适当分划如下:

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$

$$\begin{matrix} p & n-p-q & q \end{matrix}$$

```
if  q < n
    if  B22 奇异  则
        a_{n-q,n-q-1} 为零
    else
      在 A22 和 B22 上用算法 7.7.2
          A = diag(I_p, Q, I_q)^T A diag(I_p, Z, I_q)
          B = diag(I_p, Q, I_q)^T B diag(I_p, Z, I_q)
    end
  end
end
```

此算法需 $30n^3$ 个 flop. 若 $\boldsymbol{Q}$ 需求出, 还需额外的 $16n^3$ 个 flop. 若想要 $\boldsymbol{Z}$ 则还需 $20n^3$ 个 flop. 这些工作量的估计是基于每个特征值约需两个 QZ 迭代这个经验. 这样, QZ 迭代的收敛性质和 QR 迭代一样. QZ 算法的速度不受 $\boldsymbol{B}$ 秩亏损的影响.

可以证明, 计算得到的 $\boldsymbol{S}$ 和 $\boldsymbol{T}$ 满足:

$$\boldsymbol{Q}_0^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{Z}_0 = \boldsymbol{T}, \quad \boldsymbol{Q}_0^{\mathrm{T}}(\boldsymbol{B}+\boldsymbol{F})\boldsymbol{Z}_0 = \boldsymbol{S},$$

这里 $\boldsymbol{Q}_0$ 和 $\boldsymbol{Z}_0$ 为精确的正交矩阵, 而 $\|\boldsymbol{E}\|_2 \approx \mathbf{u}\|\boldsymbol{A}\|_2$ 且 $\|\boldsymbol{F}\|_2 \approx \mathbf{u}\|\boldsymbol{B}\|_2$.

**例 7.7.3** 若 QZ 算法应用到

$$\boldsymbol{A} = \begin{bmatrix} 2 & 3 & 4 & 5 & 6 \\ 4 & 4 & 5 & 6 & 7 \\ 0 & 3 & 6 & 7 & 8 \\ 0 & 0 & 2 & 8 & 9 \\ 0 & 0 & 0 & 1 & 10 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 & -1 & -1 & -1 & -1 \\ 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

则 $\boldsymbol{A}$ 的次对角元素收敛如下

| 迭 代 | $O(\|h_{21}\|)$ | $O(\|h_{32}\|)$ | $O(\|h_{43}\|)$ | $O(\|h_{54}\|)$ |
|---|---|---|---|---|
| 1 | $10^0$ | $10^1$ | $10^0$ | $10^{-1}$ |
| 2 | $10^0$ | $10^0$ | $10^0$ | $10^{-1}$ |
| 3 | $10^0$ | $10^1$ | $10^{-1}$ | $10^{-3}$ |
| 4 | $10^0$ | $10^0$ | $10^{-1}$ | $10^{-8}$ |
| 5 | $10^0$ | $10^1$ | $10^{-1}$ | $10^{-16}$ |
| 6 | $10^0$ | $10^0$ | $10^{-2}$ | 收敛 |
| 7 | $10^0$ | $10^{-1}$ | $10^{-4}$ | |
| 8 | $10^1$ | $10^{-1}$ | $10^{-8}$ | |
| 9 | $10^0$ | $10^{-1}$ | $10^{-19}$ | |
| 10 | $10^0$ | $10^{-2}$ | 收敛 | |
| 11 | $10^{-1}$ | $10^{-4}$ | | |
| 12 | $10^{-2}$ | $10^{-11}$ | | |
| 13 | $10^{-3}$ | $10^{-27}$ | | |
| 14 | 收敛 | 收敛 | | |

### 7.7.8 广义不变子空间计算

7.6 节中所讨论的许多不变子空间计算都可推广到广义特征值问题. 例如, 可以由逆迭代求近似特征向量:

> **给定** $q^{(0)} \in \mathbb{C}^{n\times n}$
> **for** $k = 1, 2, \cdots$
> 　　**解** $(A - \mu B)z^{(k)} = Bq^{(k-1)}$
> 　　**标准化:** $q^{(k)} = z^{(k)}/\|z^{(k)}\|_2$
> 　　$\lambda^{(k)} = [q^{(k)}]^{\mathrm{H}} Aq^{(k)}/[q^{(k)}]^{\mathrm{H}} Aq^{(k)}$
> **end**

当 $\boldsymbol{B}$ 为非奇异矩阵时, 这相当于将 (7.6.1) 应用于矩阵 $\boldsymbol{B}^{-1}\boldsymbol{A}$. 如果 $\mu$ 为由 QZ 算法计算出来的近似特征值, 一般只需一次迭代. 通过对 Hessenberg 三角形束进行逆迭代, QZ 迭代过程中累积 $\boldsymbol{Z}$ 的高代价可以避免.

类似于单个矩阵不变子空间的概念, 对束 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 也有降阶子空间的概念. 具体地说, 如果子空间 $\{\boldsymbol{Ax} + \boldsymbol{By} : \boldsymbol{x}, \boldsymbol{y} \in S\}$ 的维数小于或等于 $k$, 我们就说 $k$ 维子空间 $\boldsymbol{S} \subseteq \mathbb{R}^n$ 对束 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 是 "降阶" 的. 注意, 在广义 Schur 分解中, 矩阵 $\boldsymbol{Z}$ 的列确定一族降阶子空间, 这是因为如果 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n]$, $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n]$, 则 $\mathrm{span}\{\boldsymbol{Az}_1, \cdots, \boldsymbol{Az}_k\} \subseteq \mathrm{span}\,\{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k\}$ 和 $\mathrm{span}\{\boldsymbol{B}z_1, \cdots, \boldsymbol{B}z_k\} \subseteq \mathrm{span}\{q_1, \cdots, q_k\}$. 降价子空间的性质及其在扰动下的特性由 Stewert(1972) 所描述.

## 习　　题

**7.7.1** 设 $\boldsymbol{A}$ 和 $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ 且

$$U^{\mathrm{T}}BV=\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}\begin{matrix} r \\ n-r \end{matrix}, U=\begin{bmatrix} U_1 & U_2 \end{bmatrix}, \quad V=\begin{bmatrix} V_1 & V_2 \end{bmatrix}$$
$$\quad\quad r \quad n-r \quad\quad\quad r \quad n-r \quad\quad\quad r \quad n-r$$

是 $B$ 的 SVD, 其中 $D$ 是 $r\times r$ 矩阵, $r=\mathrm{rank}(B)$. 证明: 如果 $\lambda(A,B)=\mathbb{C}$, 则 $U_2^{\mathrm{T}}AV_2$ 是奇异的.

**7.7.2** 定义 $F:\mathbb{R}^n\to\mathbb{R}$ 为

$$F(x)=\frac{1}{2}\left\|Ax-\frac{x^{\mathrm{T}}B^{\mathrm{T}}Ax}{x^{\mathrm{T}}B^{\mathrm{T}}Bx}Bx\right\|_2^2$$

其中 $A$ 和 $B\in\mathbb{R}^{m\times n}$, 证明: 如果 $\nabla F(x)=0$, 则 $Ax$ 是 $Bx$ 的倍数.

**7.7.3** 设 $A$ 和 $B\in\mathbb{R}^{n\times n}$, 给出一算法来计算正交矩阵 $Q$ 和 $Z$, 使得 $Q^{\mathrm{T}}AZ$ 是上 Hessenberg 矩阵且 $Z^{\mathrm{T}}BQ$ 是上三角形矩阵.

**7.7.4** 设

$$A=\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad B=\begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix},$$

其中 $A_{11},B_{11}\in\mathbb{R}^{k\times k}$ 和 $A_{22},B_{22}\in\mathbb{R}^{j\times j}$. 在什么情形下存在

$$X=\begin{bmatrix} I_k & X_{12} \\ 0 & I_j \end{bmatrix}, \quad Y=\begin{bmatrix} I_k & Y_{12} \\ 0 & I_j \end{bmatrix},$$

使得 $Y^{-1}AX$ 和 $Y^{-1}BX$ 均为块对角矩阵? 这是一个广义Sylvester方程问题. 当 $A_{11}$, $A_{22}$, $B_{11}$, $B_{22}$ 为上三角形矩阵时的情形时, 试给出一特殊算法. 见 Kågström(1994).

**7.7.5** 设 $\mu\notin\lambda(A,B)$. 试给出 $A_1=(A-\mu B)^{-1}A$ 和 $B_1=(A-\mu B)^{-1}B$ 的特征值和特征向量与 $A-\lambda B$ 的广义特征值和特征向量之关系.

**7.7.6** 设 $A,B,C,D\in\mathbb{R}^{n\times n}$, 说明怎样计算正交矩阵 $Q,Z,U,V$, 使得 $Q^{\mathrm{T}}AU$ 是上 Hessenberg 矩阵和 $V^{\mathrm{T}}CZ, Q^{\mathrm{T}}BV$ 及 $V^{\mathrm{T}}DZ$ 均为上三角形矩阵. 注意到这将束 $AC-\lambda BD$ 化为 Hessenberg 三角型. 你的算法不要显式形成乘积 $AC$ 或 $BD$, 也不用计算任何矩阵的逆. 见 Van Loan(1975).

## 本节注释与参考文献

广义特征值问题的数学性质可见下文:

F. Gantmacher(1959). *The Theory of Matrices,* vol.2, Chelsea,New York.

H. W. Turnbill and A. C. Aitken(1961). *An Introduction to the Theory of Canonical Matrices,* Dover, New York.

I. Erdelyi(1967). "On the Matrix Equation $Ax=\lambda Bx$," *J.Math. Anal. and Applic.* 17, 119–132

一本很好的包括了 $A-\lambda B$ 问题诸多方面的综合性专著是:

B. Kågström and A. Ruhe(1983). *Matrix Pencils,* Proc. Pite Havsbad, 1982, Lecture Notes in Mathematics 973, Springer-Verlag, New York and Berlin.

下面文章处理广义特征值问题的扰动理论：

G. W. Stewart(1972). "On the Sensitivity of the Eigenvalue Problem $Ax = \lambda Bx$," *SIAMJ. Num. Anal.* 9, 669–686.

G. W. Stewart(1973). "Error and Perturbation Bounds for Subspaces Associated with Cetain Eigenvalue Problems," *SIAM Review* 15, 727–764.

G. W. Stewart(1975). "Gershgorin Theory for the Generalized Eigenvalue Problem $Ax = \lambda Bx$," *Math. Comp.* 29, 600–606.

G. W. Stewart(1978). "Perturbation Theory for the Generalized Eigenvalue Problem," in *Recent Advances in Numerical Analysis,* ed C. de Boor and G. H. Golub, Academic Press, New York.

A. Pokrzywa(1986). "On Perturbations and the Equivalence Orbit of a Matrix Pencil," *Lin. Alg. and Applic.* 82, 99–121.

与 QZ 有关的文章有：

C. B. Moler and G. W. Stewart(1973). "An Algorithm for Generalized Matrix Eigenvalue Problems," *SIAM J. Num. Anal.* 10, 241–256.

L. Kaufman(1974). "The LZ Algorithm to Solve the Generalized Eigenvalue Problem," *SIAM J. Num. Anal.* 11, 997–1024.

R. C. Ward(1975). "The Combination Shift QZ Algorithm," *SIAM J. Num. Anal.* 12, 835–853.

C. F. Van Loan(1975). " A General Matrix Eigenvalue Algorithm," *SIAN J. Num. Anal.* 12, 819–834.

L. Kaufman(1977). "Some Thoughts on the QZ Algorithm for Solving the Generalized Eigenvalue Problem," *ACM Trans. Math. Soft.* 3, 65–75.

R. C. Ward(1981). "Balancing the Generalized Eigenvalue Problem," *SIAM J. Sci and Stat. Comp.* 2, 141–152.

P. Van Dooren(1982). "Algorithm 590: DSUBSP and EXCHQZ: Fortran Routines for Computing Deflating Subspaces with Specified Spectrum," *ACM Trans. Math. Software* 8, 376–382.

D. Watkins and L. Elsner(1994). "Theory of Decomposition and Bulge-Chasing Algorithms for the Generalized Eigenvalue Problem," *SIAM J. Matrix Anal. Appl.* 15, 943–967.

与 Hessenberg 分解一样，作为 QZ 前端的 Hessenberg 三角形分解就其本身而也是很重要的. 见：

W. Enright and S. Serbin(1978). "A Note on the Efficient Solution of Matrix Pencil Systems," *BIT* 18, 276–281.

另外的求解框架下文中提出：

V. N. Kublanovskaja and V. N. Fadeeva(1964). "Computational Methods for the Solution of a Generalized Eigenvalue Problem," *Amer. Math. Soc. Transl.* 2, 271–290.

G. Peters and J. H. Wilkinson(1970a). "$Ax = \lambda Bx$ and the Generalized Eigenproblem," *SIAM J. Num. Anal.* 7, 479–492.

G. Rodrigue(1973). "A Gradient Method for the Matrix Eigenvalue Problem $Ax = \lambda Bx$," *Numer. Math.* 22, 1–16.

H. R. Schwartz(1974). "The Method of Coordinate Relaxation for $(A - \lambda B)x = 0$," *Num. Math.* 23, 135–152.

A. Jennings and M. R. Osborne(1977). "Generalized Eigenvalue Problems for Certain Unsymmetric Band Matrices," *Lin. Alg. and Its Applic.* 29, 139–150.

V. N. Kublanovskaya(1984). "AB Algorithm and Its Modifications for the Spectral Problem of Linear Pencils of Matrices," *Numer. Math.* 43, 329–342.

C. Oara(1994). "Proper Deflating Subspaces: Properties, Algorithms, and Applications," *Numerical Algorithms* 7, 355–373.

广义 $Ax = \lambda Bx$ 问题在一些重要的控制论应用方面很重要. 见:

P. Van Dooren(1981). "A Generalized Eigenvalue Approach for Solving Riccati Equations," *SIAM J. Sci. and Stat. Comp.* 2, 121–135.

P. Van Dooren(1981). "The Generalized Eigenstructure Problem in Linear System Theory," *IEEE Trans. Auto. Cont.* AC-26, 111–128.

W. F. Arnold and A. J. Laub(1984). "Generalized Eigenproblem Algorithms and Software for Algebraic Riccati Equations," *Proc. IEEE* 72, 1746–1754.

J. W. Demmel and B. Kågström(1988). "Accurate Solutions of Ill-Posed Problems in Control Theory," *SIAM J.Matrix Anal. Appl* 126–145.

U. Flaschka,W-W. Li, and J-L. Wu(1992). "A KQZ Algorithm for Solving Linear-Response Eigenvalue Equations," *Lin. Alg. and its Applic.* 165,93–123.

长方形广义特征值问题在某些应用中出现. 见:

G. L. Thompson and R. L. Weil(1970). "Reducing the Rank of $A - \lambda B$," *Proc, Amer. Math. Sec.* 26, 548–554.

G. L. Thompson and R. L. Weil(1972). "Roots of Matrix Pencils $Ay = \lambda By$: Existence, Calculations, and Relations to Game Theory," *Lin. Alg. and Its Applic.* 5, 207–226.

G. W. Stewart(1994). "Perturbation Theory for Rectangular Matrix Pencils," *Lin. Alg. and Applic.* 208/209, 297–301.

束 $A - \lambda B$ 的 Kronecker 结构类似于 $A - \lambda I$ 的 Jordan 结构: 它为有关应用提供很有用的信息.

J. H. Wilkinson(1978). "Linear Differential Equations and Kronecker's Canonical Form," in *Recent Advances in Numerical Analysis,* ed C.de Boor and G. H. Golub, Academic Press, New York, pp. 231–265.

对 Kronecker 结构的研究产生了大量的新算法和分析结果.

J. H. Wilkinson(1979). "Kronecker's Canonical Form and the QZ Algorithm," *Lin. Alg. and Its Applic.* 28, 285–303.

P. Van Dooren(1979). "The Computation of Kronecker's Canonical Form of a Singular Pencil," *Lin. Alg. and Its Applic.* 27, 103–140.

J. W. Demmel(1983). "The Condition Number of Equivalence Transformations that Block Diagonalize Matrix Pencils," *SIAM J. Numer. Anal.* 20, 599–610.

J. W. Demmel and B. Kågström(1978). "Computing Stable Eigendecompositions of Matrix Pencils," *Linear Alg. and Its Applic* 88/89, 139–186.

B. Kågström(1985). "The Generalized Singular Value Decomposition and the General $A-\lambda B$ Problem," *BIT* 24, 568–583.

B. Kågström(1986). "RGSVD: An Algorithm for Computing the Kronecker Structure and Reducing Subspaces of Singular $A - \lambda B$ Pencils," *SIAM J. Sci.and Stat.Comp.* 7, 185–211.

J. Demmel and B. Kågström(1986). "Stably Computing the Kronecker Structure and Reducing Subspaces of Singular Pencils $A-\lambda B$ for Uncertain Data," in *Large Scale Eigenvalue Problems,* J.Cullum and R.A.Willoughby(eds), North-Holland, Amsterdam.

T. Beelen and P. Van Dooren(1988). "An Improved Algorithm for the Computation of Kronecker's Canonical Form of a Singular Pencil," *Lin. Alg. and Its Applic.* 105, 9–65.

B. Kågström and L. Westin(1989). "Generalized Schur Methods with Condition Estimators for Solving the Generalized Sylvester Equation," *IEEE Trans. Auto. Cont.* AC-34, 745–751.

B. Kågström and P. Poromaa(1992). "Distributed and Shared Memory Block Algorithms for the Triangular Sylvester Equation with $\text{sep}^{-1}$ Estimators," *SIAM J. Matrix Anal. Appl.* 13, 90–101.

B. Kågström(1994). "A Perturbation Analysis of the Generalized Sylvester Equation $(AR-LB, DR-LE)=(C,F)$," *SIAM J. Matrix Anal. Appl.* 15, 1045–1060.

E. Elmroth and B. Kågström(1996). "The Set of 2-by-3 Matrix Pencils-Kronecker Structure and their Transitions under Perturbations," *SIAM J. Matrix Anal.*, to appear.

A. Edelman, E. Elmroth, and B. Kågström(1996). "A Geometric Approach to Perturbation Theory of Matrices and Matrix Pencils," *SIAM J. Matrix Anal., to* appear.

# 第 8 章　对称特征值问题

具有丰富数学结构的对称特征值问题是数值线性代数中最漂亮的问题之一. 首先我们简要讨论对称矩阵的数学性质, 它们奠定了此计算的基础. 在 8.2 节和 8.3 节中我们提出了多种幂迭代法, 最后集中讨论对称 QR 算法.

在 8.4 节中我们讨论 Jacobi 方法, 这是文献中出现最早的矩阵算法之一. 由于易于并行计算和在某种条件下具有高精度, 这种方法近来又引起人们的兴趣.

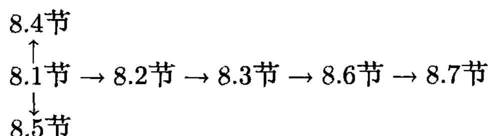
在 8.5 节中对三对角情形给出了不同的方法, 包括二分法和分而治之技巧.

在 8.6 节中详述怎样计算奇异值分解. 核心算法是对称 QR 迭代法应用于双对角矩阵的变形.

在最后一节我们就 $\boldsymbol{A}$ 为对称矩阵且 $\boldsymbol{B}$ 是对称正定矩阵时这一重要情形讨论广义特征值问题 $\boldsymbol{Ax} = \lambda \boldsymbol{Bx}$. 对这个特殊结构的广义特征值问题, 没有适当的类似基于正交的 QZ 算法 (见 7.7 节) 存在. 然而, 有几个成功的方法可以用, 我们给出这些方法, 并讨论广义奇异值分解.

**预备知识**

阅读本章需要掌握第 1 章, 2.1 节 ~2.5 节和 2.7 节, 第 3 章, 4.1 节 ~4.3 节, 5.1 节 ~5.5 节和 7.1.1 节的知识. 本章各节间的关系如下:




本章的许多算法和理论与第 7 章非对称的部分相对应. 然而, 除了少量概念和定义外, 我们处理对称矩阵的特征值问题的方法在读第 7 章之前就能学习.

补充参考文献包括 Wilkinson(1965), Stewart(1973), Gourlay and Watson(1973), Hager(1988), Chatelin(1993), Parlett(1980), Stewart and Sun(1990), Watkins(1991), Jennings and McKeowen(1992) 以及 Datta(1995). 对本章很重要的 MATLAB函数是schur和svd. 与 LAPACK 相关的程序如下.

| LAPACK: 对称特征值问题 | |
|---|---|
| _SYEV | 所有特征值和特征向量 |
| _SYEVD | 同上但用分而治之求特征向量 |
| _SYEVX | 部分特征值和特征向量 |
| _SYTRD | Householder 三对角化 |
| _SBTRD | Householder 三对角化 (带状 $\boldsymbol{A}$) |
| _SPTRD | Householder 三对角化 ($\boldsymbol{A}$ 打包存储) |

(续)

| LAPACK：对称特征值问题 | |
|---|---|
| _STEQR | 三对角矩阵的所有特征值和特征向量 (隐 QR) |
| _STEDC | 三对角矩阵的所有特征值和特征向量 (分而治之) |
| _STERF | 三对角矩阵的所有特征值 (无求根 QR) |
| _PTEQR | 正定三对角矩阵的所有特征值和特征向量 |
| _STEBZ | 三对角矩阵的部分特征值 (对分法) |
| _STEIN | 三对角矩阵的部分特征值 (逆迭代) |

| LAPACK：对称正定特征值问题 | |
|---|---|
| _SYGST | 将 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 化为 $\boldsymbol{C}-\lambda\boldsymbol{I}$ 形式 |
| _PBSTF | 分裂 Cholesky 分解 |
| _SBGST | 用分裂 Cholesky 分解将带状 $\boldsymbol{A}-\lambda\boldsymbol{B}$ 化为 $\boldsymbol{C}-\lambda\boldsymbol{I}$ 形式 |

| LAPACK：SVD | |
|---|---|
| _GESVD | $\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$ |
| _BDSQR | 实双对角矩阵的 SVD |
| _GEBRD | 一般矩阵的双对角化 |
| _ORGBR | 产生正交变换 |
| _GBBRD | 带状矩阵的双对角化 |

| LAPACK：广义奇异值问题 | |
|---|---|
| _GGSVP | 将 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}-\mu^2\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$ 化为三角形 $\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1-\mu^2\boldsymbol{B}_1^{\mathrm{T}}\boldsymbol{B}_1$ |
| _TGSJA | 计算两个三角形矩阵的 GSVD |

## 8.1　性质与分解

在这一节, 我们陈述一些所需的数学知识, 以便提出和分析对称特征值算法.

### 8.1.1　特征值和特征向量

对称性保证了 $\boldsymbol{A}$ 的所有特征值都是实的且有一组由特征向量组成的正交基.

**定理 8.1.1 (对称 Schur 分解)**　*如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称的, 则存在一个实正交矩阵 $\boldsymbol{Q}$ 使得*

$$\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_1,\cdots,\lambda_n)$$

*而且, 对 $k=1:n$, $\boldsymbol{A}Q(:,k)=\lambda_kQ(:,k)$. 见定理 7.1.3.*

**证明**　设 $\lambda_1\in\lambda(\boldsymbol{A})$ 且 $x\in\mathbb{C}^n$ 是 2 范数单位特征向量, $\boldsymbol{A}\boldsymbol{x}=\lambda_1\boldsymbol{x}$. 由于 $\lambda_1=\boldsymbol{x}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{x}=\boldsymbol{x}^{\mathrm{H}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{x}=\overline{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{x}}=\bar{\lambda}_1$, 即知 $\lambda_1\in\mathbb{R}$. 这样, 可以假定 $\boldsymbol{x}\in\mathbb{R}^n$. 令 $\boldsymbol{P}_1\in\mathbb{R}^{n\times n}$ 是一个 Householder 矩阵, 使得 $\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{e}_1=I_n(:,1)$. 由于 $\boldsymbol{A}\boldsymbol{x}=\lambda_1\boldsymbol{x}$ 可知 $(\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_1)e_1=\lambda e_1$. 这是说 $\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_1$ 的第一列是 $e_1$ 的倍数. 但因 $\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_1$ 是对称矩阵, 故它必有形式

$$\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_1=\begin{bmatrix}\lambda_1 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{A}_1\end{bmatrix},$$

其中 $A_1 \in \mathbb{R}^{(n-1)\times(n-1)}$ 是对称矩阵. 由归纳法我们假定有一个正交矩阵 $Q_1 \in \mathbb{R}^{(n-1)\times(n-1)}$ 使得 $Q_1^{\mathrm{T}} A_1 Q_1 = \Lambda_1$ 是对角矩阵. 令 $Q = P_1 \begin{bmatrix} 1 & 0 \\ 0 & Q_1 \end{bmatrix}$ 和 $\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \Lambda_1 \end{bmatrix}$ 并且比较矩阵方程 $AQ = Q\Lambda$ 的列就可得到定理. □

**例 8.1.1** 如果

$$A = \begin{bmatrix} 6.8 & 2.4 \\ 2.4 & 8.2 \end{bmatrix} \quad 和 \quad Q = \begin{bmatrix} 0.6 & -0.8 \\ 0.8 & 0.6 \end{bmatrix},$$

则 $Q$ 为正交矩阵且 $Q^{\mathrm{T}}AQ$=diag(10, 5).

我们将用记号 $\lambda_k(A)$ 表示对称矩阵 $A$ 的第 $k$ 个最大特征值, 这样,

$$\lambda_n(A) \leqslant \cdots \leqslant \lambda_2(A) \leqslant \lambda_1(A).$$

由 2 范数的正交不变性可以知道 $A$ 有奇异值 $\{|\lambda_1(A)|, \cdots, |A_n(A)|\}$ 且

$$\|A\|_2 = \max\{|\lambda_1(A)|, |\lambda_n(A)|\}.$$

对称矩阵的特征值有一个"极小化极大"特征, 它是基于可看成是二次型比值 $x^{\mathrm{T}}Ax/x^{\mathrm{T}}x$ 的量.

**定理 8.1.2 (Courant-Fischer 极小化极大定理)** 如果 $A \in \mathbb{R}^{n\times n}$ 为对称矩阵, 则

$$\lambda_k(A) = \max_{\dim(S)=k} \min_{0\neq y\in S} \frac{y^{\mathrm{T}}Ay}{y^{\mathrm{T}}y}, \quad 对 k = 1 : n.$$

**证明** 令 $Q^{\mathrm{T}}AQ = \mathrm{diag}(\lambda_i)$ 是 Schur 分解, 其中 $\lambda_k = \lambda_k(A)$, 且 $Q = [q_1, q_2, \cdots, q_n]$. 定义

$$S_k = \mathrm{span}\{q_1, \cdots, q_k\}$$

为与 $\lambda_1, \cdots, \lambda_k$ 相对应的不变子空间, 容易证明

$$\max_{\dim(S)=k} \min_{0\neq y\in S} \frac{y^{\mathrm{T}}Ay}{y^{\mathrm{T}}y} \geqslant \min_{0\neq y\in S_k} \frac{y^{\mathrm{T}}Ay}{y^{\mathrm{T}}y} = q_k^{\mathrm{T}}Aq_k = \lambda_k(A).$$

现证明反过来的不等式. 令 $S$ 是任意 $k$ 维子空间且注意到它一定与维数为 $n-k+1$ 的子空间 $\mathrm{span}\{q_k, \cdots, q_n\}$ 相交. 设 $y_* = \alpha_k q_k + \cdots + \alpha_n q_n$ 在此交集中, 则

$$\min_{0\neq y\in S} \frac{y^{\mathrm{T}}Ay}{y^{\mathrm{T}}y} \leqslant \frac{y_*^{\mathrm{T}}Ay_*}{y_*^{\mathrm{T}}y_*} \leqslant \lambda_k(A).$$

由于此不等式对所有 $k$ 维子空间成立, 故

$$\max_{\dim(S)=k} \min_{0\neq y\in S} \frac{y^{\mathrm{T}}Ay}{y^{\mathrm{T}}y} \leqslant \lambda_k(A).$$

因此定理得证. □

如果 $A \in \mathbb{R}^{n\times n}$ 是对称正定矩阵, 则 $\lambda_n(A) > 0$.

### 8.1.2　特征值的敏感度

对称特征值问题的一个重要的求解框架包括产生一列正交变换 $\{Q_k\}$, 这些矩阵使得 $Q_k^{\mathrm{T}}AQ_k$ 逐步"更加对角化". 问题自然而然产生, 一个矩阵的对角元与它的特征值究竟近似到什么程度?

**定理 8.1.3 (Gershgorin)**　设 $A \in \mathbb{R}^{n\times n}$ 为对称矩阵且 $Q \in \mathbb{R}^{n\times n}$ 为正交矩阵. 若 $Q^{\mathrm{T}}AQ = D + F$, 其中 $D = \mathrm{diag}(d_1, \cdots, d_n)$, 且 $F$ 有零对角元, 则

$$\lambda(A) \subseteq \bigcup_{i=1}^{n}[d_i - r_i, d_i + r_i]$$

其中对 $i = 1 : n$, $r_i = \sum_{i=1}^{n}|f_{ij}|$. 见定理 7.2.1.

**证明**　设 $\lambda \in \lambda(A)$, 且不失一般性假定对 $i = 1 : n$ 有 $\lambda \neq d_i$. 由于 $(D-\lambda I)+F$ 奇异, 由引理 2.3.3 可知, 对某个 $k(1 \leqslant k \leqslant n)$, 有

$$1 \leqslant \|(D - \lambda I)^{-1}F\|_\infty = \sum_{j=1}^{n}\frac{|f_{kj}|}{|d_k - \lambda|} = \frac{r_k}{|d_k - \lambda|}.$$

但这隐含 $\lambda \in [d_k - r_k, d_k + r_k]$. □

**例 8.1.2**　矩阵

$$A = \begin{bmatrix} 2.0000 & 0.1000 & 0.2000 \\ 0.2000 & 5.0000 & 0.3000 \\ 0.1000 & 0.3000 & -1.0000 \end{bmatrix}$$

有 Gershgorin 区间 [1.7, 2.3], [4.5, 5.5] 和 [−1.4, −6] 以及特征值 1.9984, 5.0224 和 −1.0208.

下面的结果表明, 如果 $A$ 被一个对称矩阵 $E$ 所扰动, 则它的特征值变化范围不超过 $\|E\|$.

**定理 8.1.4 (Wielandt-Hoffman)**　若 $A$ 和 $A+E$ 是 $n\times n$ 对称矩阵, 则

$$\sum_{i=1}^{n}(\lambda_i(A+E) - \lambda_i(A))^2 \leqslant \|E\|_F^2.$$

**证明**　在 Wilkinson(1965, 104~108 页) 或 Stewart and Sun(1991, 189~191 页) 中均能找到证明. 也可参看习题 8.1.5. □

**例 8.1.3**　如果

$$A = \begin{bmatrix} 6.8 & 2.4 \\ 2.4 & 8.2 \end{bmatrix} \quad 和 \quad E = \begin{bmatrix} 0.002 & 0.003 \\ 0.003 & 0.001 \end{bmatrix},$$

则 $\lambda(A) = \{5, 10\}$ 和 $\lambda(A+E) = \{4.9988, 10.004\}$, 满足

$$\begin{aligned} 1.95 \times 10^{-5} = |4.9988 - 5|^2 + |10.004 - 10|^2 &\leqslant \|E\|_F^2 \\ &= 2.3 \times 10^{-5}. \end{aligned}$$

**定理 8.1.5** 如果 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{E}$ 是 $n\times n$ 对称矩阵, 则对 $k=1:n$ 有

$$\lambda_k(\boldsymbol{A})+\lambda_n(\boldsymbol{E})\leqslant\lambda_k(\boldsymbol{A}+\boldsymbol{E})\leqslant\lambda_k(\boldsymbol{A})+\lambda_1(\boldsymbol{E}).$$

**证明** 这可由极小化极大定理证明. 参看 Wilkinson(1965, 101~102 页) 或 Stewart and Sun(1990, 203 页). □

**例 8.1.4** 如果

$$\boldsymbol{A}=\begin{bmatrix}6.8 & 2.4\\ 2.4 & 8.2\end{bmatrix} \quad 和 \quad \begin{bmatrix}0.002 & 0.003\\ 0.003 & 0.001\end{bmatrix},$$

则 $\lambda(\boldsymbol{A})=\{5,10\},\lambda(\boldsymbol{E})=\{-0.0015,0.0045\}$, 且 $\lambda(\boldsymbol{A}+\boldsymbol{E})=\{4.9988,10.0042\}$, 满足

$$5-0.0015\leqslant4.9988\leqslant5+0.0045$$

$$10-0.0015\leqslant10.0042\leqslant10+0.0045$$

**推论 8.1.6** 若 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{E}$ 是 $n\times n$ 对称矩阵, 则对 $k=1:n$ 有

$$|\lambda_k(\boldsymbol{A}+\boldsymbol{E})-\lambda_k(\boldsymbol{A})|\leqslant\|\boldsymbol{E}\|_2.$$

**证明** $|\lambda_k(\boldsymbol{A}+\boldsymbol{E})-\lambda_k(\boldsymbol{A})|\leqslant\max\{|\lambda_n(\boldsymbol{E})|,|\lambda_1(\boldsymbol{E})|\}=\|\boldsymbol{E}\|_2.$ □

从极小化极大性质可得出几个更有用的扰动结论.

**定理 8.1.7 (交错性质)** 如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称矩阵且 $A_r=(1:r,1:r)$, 则对 $r=1:n-1$ 有

$$\lambda_{r+1}(A_{r+1})\leqslant\lambda_r(A_r)\leqslant\lambda_r(A_{r+1})\leqslant\cdots\leqslant\lambda_2(A_{r+1})\leqslant\lambda_1(A_r)\leqslant\lambda_1(A_{r+1}).$$

**证明** Wilkinson(1965, 103~104 页). □

**例 8.1.5** 如果

$$\boldsymbol{A}=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 2 & 3 & 4\\ 1 & 3 & 6 & 10\\ 1 & 4 & 10 & 20\end{bmatrix}$$

则 $\lambda(\boldsymbol{A}_1)=\{1\},\lambda(\boldsymbol{A}_2)=\{0.3820,2.6180\},\lambda(\boldsymbol{A}_3)=\{0.1270,1.0000,7.873\}$, 且 $\lambda(\boldsymbol{A}_4)=\{0.0380,0.4538,2.2034,26.3047\}$.

**定理 8.1.8** 设 $\boldsymbol{B}=\boldsymbol{A}+\tau\boldsymbol{c}\boldsymbol{c}^{\mathrm{T}}$, 其中 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称矩阵, $\boldsymbol{c}\in\mathbb{R}^n$ 是 2 范数单位向量且 $\tau\in\mathbb{R}$. 若 $\tau\geqslant0$, 则

$$\lambda_i(\boldsymbol{B})\in[\lambda_i(\boldsymbol{A}),\lambda_{i-1}(\boldsymbol{A})],\quad i=2:n.$$

然而若 $\tau\leqslant0$, 则

$$\lambda_i(\boldsymbol{B})\in[\lambda_{i+1}(\boldsymbol{A}),\lambda_i(\boldsymbol{A})],\quad i=1:n-1.$$

在任一情形下都存在非负数 $m_1,\cdots,m_n$ 使得

$$\lambda_i(\boldsymbol{B})=\lambda_i(\boldsymbol{A})+m_i\tau,\quad i=1:n$$

而且 $m_1+\cdots+m_n=1$.

**证明** Wilkinson(1965, 94~97 页), 也可参见习题 8.1.8. □

### 8.1.3 不变子空间

许多特征值计算过程都是将原来问题分解成许多小的子问题求解. 下面的定理是这种求解框架的基石.

**定理 8.1.9** 假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵且

$$\boldsymbol{Q} = [\ \underset{r}{\boldsymbol{Q}_1} \quad \underset{n-r}{\boldsymbol{Q}_2}\ ]$$

为正交矩阵. 如果 $\text{ran}(\boldsymbol{Q}_1)$ 是不变子空间, 则

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{D} = \begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_2 \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}, \quad \begin{matrix} r \quad n-r \end{matrix} \tag{8.1.1}$$

且 $\lambda(\boldsymbol{A}) = \lambda(\boldsymbol{D}_1) \cup \lambda(\boldsymbol{D}_2)$. 参看引理 7.1.2.

**证明** 如果

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{E}_{21}^{\mathrm{T}} \\ \boldsymbol{E}_{21} & \boldsymbol{D}_2 \end{bmatrix},$$

则从 $\boldsymbol{AQ} = \boldsymbol{QD}$, 我们有 $\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{D}_1 = \boldsymbol{Q}_2\boldsymbol{E}_{21}$. 由于 $\text{ran}(\boldsymbol{Q}_1)$ 是不变子空间, $\boldsymbol{Q}_2\boldsymbol{E}_{21}$ 的列向量也在 $\text{ran}(\boldsymbol{Q}_1)$ 中, 因此与 $\boldsymbol{Q}_2$ 的列向量垂直, 故

$$\boldsymbol{0} = \boldsymbol{Q}_2^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{D}_1) = \boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{Q}_2\boldsymbol{E}_{21} = \boldsymbol{E}_{21}.$$

所以, (8.1.1) 成立. 易证

$$\det(\boldsymbol{A} - \lambda\boldsymbol{I}_n) = \det(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} - \lambda\boldsymbol{I}_n) = \det(\boldsymbol{D}_1 - \lambda\boldsymbol{I}_r)\det(\boldsymbol{D}_2 - \lambda\boldsymbol{I}_{n-r}),$$

这就证实了 $\lambda(\boldsymbol{A}) = \lambda(\boldsymbol{D}_1) \cup \lambda(\boldsymbol{D}_2)$. □

不变子空间的扰动敏感度依赖于相关特征值与谱中的其他特征值的分离度. 两个对称矩阵 $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 的特征值的分离度的适当度量由下式给出:

$$\text{sep}(\boldsymbol{B}, \boldsymbol{C}) = \min_{\substack{\lambda \in \lambda(\boldsymbol{B}) \\ \mu \in \lambda(\boldsymbol{C})}} |\lambda - \mu|. \tag{8.1.2}$$

由此定义, 我们有下述定理.

**定理 8.1.10** 设 $\boldsymbol{A}$ 和 $\boldsymbol{A} + \boldsymbol{E}$ 是 $n \times n$ 对称矩阵且

$$\boldsymbol{Q} = [\ \underset{r}{\boldsymbol{Q}_1} \quad \underset{n-r}{\boldsymbol{Q}_2}\ ]$$

是一个正交矩阵, 它使得 $\text{ran}(\boldsymbol{Q}_1)$ 是 $\boldsymbol{A}$ 的不变子空间, 将矩阵 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$ 和 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{E}\boldsymbol{Q}$ 划分如下:

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{D}_1 & 0 \\ 0 & \boldsymbol{D}_2 \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix} \qquad \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{E}\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{E}_{11} & \boldsymbol{E}_{21}^{\mathrm{T}} \\ \boldsymbol{E}_{21} & \boldsymbol{E}_{22} \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$$
$$\quad r \quad n-r \qquad\qquad\qquad\qquad r \quad n-r$$

如果 $\text{sep}(\boldsymbol{D}_1, \boldsymbol{D}_2) > 0$ 且

$$\|\boldsymbol{E}\|_2 \leqslant \frac{\text{sep}(\boldsymbol{D}_1, \boldsymbol{D}_2)}{5},$$

则存在一个满足

$$\|P\|_2 \leqslant \frac{4}{\text{sep}(D_1, D_2)}\|E_{21}\|_2$$

的矩阵 $P \in \mathbb{R}^{(n-r)\times r}$ 使得 $\hat{Q}_1 = (Q_1 + Q_2P)(I + P^{\mathrm{T}}P)^{-1/2}$ 的列定义了 $A+E$ 的不变子空间的一组正交基, 参看定理 7.2.4.

**证明** 此定理是 Stewart(1973) 中定理 4.11 的稍微改动, 矩阵 $(I + P^{\mathrm{T}}P)^{-1/2}$ 是 $(I + P^{\mathrm{T}}P)$ 的平方根的逆. 见 4.2.10 节. □

**推论 8.1.11** 若定理的条件成立, 则

$$\text{dist}(\text{ran}(Q_1), \text{ran}(\hat{Q}_1)) \leqslant \frac{4}{\text{sep}(D_1, D_2)}\|E_{21}\|_2.$$

参见推论 7.2.5.

**证明** 用奇异值分解 (SVD) 可证

$$\|P(I + P^{\mathrm{T}}P)^{-1/2}\|_2 \leqslant \|P\|_2. \tag{8.1.3}$$

由 $Q_2^{\mathrm{T}}\hat{Q}_1 = P(I + P^{\mathrm{H}}P)^{-1/2}$ 可得

$$\begin{aligned}\text{dist}(\text{ran}(Q_1), \text{ran}(\hat{Q}_1)) &= \|Q_2^{\mathrm{T}}\hat{Q}_1\|_2 = \|P(I + P^{\mathrm{H}}P)^{-1/2}\|_2 \\ &\leqslant \|P\|_2 \leqslant 4\|E_{21}\|_2/\text{sep}(D_1, D_2).\end{aligned}$$ □

这样, $\text{sep}(D_1, D_2)$ 的倒数可认为是度量不变子空间 $\text{ran}(Q_1)$ 的敏感度的条件数.

扰动对单特征向量的影响非常重要, 对此重要情形我们强调以上结论.

**定理 8.1.12** 设 $A$ 和 $A+E$ 是 $n \times n$ 对称矩阵且

$$Q = \underset{\quad 1 \qquad n-1}{[\, q_1 \quad Q_2 \,]}$$

是正交矩阵, 其中 $q_1$ 是 $A$ 的特征向量. 将 $Q^{\mathrm{T}}AQ$ 和 $Q^{\mathrm{T}}EQ$ 划分如下:

$$Q^{\mathrm{T}}AQ = \begin{bmatrix} \lambda & 0 \\ 0 & D_2 \end{bmatrix}\begin{matrix} 1 \\ n-1 \end{matrix}, \quad Q^{\mathrm{T}}EQ = \begin{bmatrix} \varepsilon & e^{\mathrm{T}} \\ e & E_{22} \end{bmatrix}\begin{matrix} 1 \\ n-1 \end{matrix}$$
$$\qquad 1 \quad n-1 \qquad\qquad\qquad 1 \quad n-1$$

如果 $d = \min\limits_{\mu \in \lambda(D_2)} |\lambda - \mu| > 0$ 且

$$\|E\|_2 \leqslant \frac{d}{4},$$

则存在满足

$$\|p\|_2 \leqslant \frac{4}{d}\|e\|_2.$$

的 $p \in \mathbb{R}^{n-1}$ 使得 $\hat{q}_1 = (q_1 + Q_2p)/\sqrt{1 + p^{\mathrm{T}}p}$ 是 $A+E$ 的 2 范数单位特征向量. 而且,

$$\text{dist}(\text{span}\{q_1\}, \text{span}\{\hat{q}_1\}) = \sqrt{1 - (q_1^{\mathrm{T}}\hat{q}_1)^2} \leqslant \frac{4}{d}\|e\|_2.$$

也参见推论 7.2.6.

**证明** 应用定理 8.1.10 和推论 8.1.11($r=1$) 并且注意到如果 $D_1=(\lambda)$ 则 $d=\text{sep}(D_1,D_2)$. □

**例 8.1.6** 如果 $\boldsymbol{A}=\text{diag}(0.999,1.001,2.0)$ 和

$$\boldsymbol{E}=\begin{bmatrix}0.00 & 0.01 & 0.01\\ 0.01 & 0.00 & 0.01\\ 0.01 & 0.01 & 0.00\end{bmatrix},$$

则 $\hat{\boldsymbol{Q}}^{\text{T}}(\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{Q}}=\text{diag}(0.9899,1.0098,2.0002)$, 其中

$$\hat{\boldsymbol{Q}}=\begin{bmatrix}-0.7418 & 0.6706 & 0.0101\\ 0.6708 & 0.7417 & 0.0101\\ 0.0007 & -0.0143 & 0.9999\end{bmatrix}$$

是正交矩阵, 令 $\hat{\boldsymbol{q}}_i=\hat{\boldsymbol{Q}}\boldsymbol{e}_i, i=1,2,3$. 于是, $\hat{\boldsymbol{q}}_i$ 是 $\boldsymbol{A}$ 的特征向量 $\boldsymbol{q}_i=\boldsymbol{e}_i$ 的扰动. 计算表明

$$\text{dist}\{\text{span}\{\boldsymbol{q}_1\},\text{span}\{\hat{\boldsymbol{q}}_1\}\}=\text{dist}\{\text{span}\{\boldsymbol{q}_2\},\text{span}\{\hat{\boldsymbol{q}}_2\}\}=0.67.$$

这样, 因为特征向量 $\boldsymbol{q}_1$ 和 $\boldsymbol{q}_2$ 对应于附近的特征值, 故不能精确计算. 另一方面, 由于 $\lambda_1$ 和 $\lambda_2$ 与 $\lambda_3$ 分开, 它们定义的二维子空间不是特别敏感, 这从 $\text{dist}\,\{\text{span}\{q_1,q_2\},\text{span}\{\hat{q}_1,\hat{q}_2\}\}=0.01$ 可以看出.

### 8.1.4 近似不变子空间

如果 $\boldsymbol{Q}_1\in\mathbb{R}^{n\times r}$ 的列线性无关并且对某 $\boldsymbol{S}\in\mathbb{R}^{r\times r}$余量矩阵$\boldsymbol{R}=\boldsymbol{A}\boldsymbol{Q}_1-\boldsymbol{Q}_1\boldsymbol{S}$ 较小, 则 $\boldsymbol{Q}_1$ 的列向量定义了一个近似不变子空间. 当有了这样一个矩阵时, 让我们看看 $\boldsymbol{A}$ 的特征系统有何特点.

**定理 8.1.13** 设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 和 $\boldsymbol{S}\in\mathbb{R}^{r\times r}$ 是对称矩阵且

$$\boldsymbol{A}\boldsymbol{Q}_1-\boldsymbol{Q}_1\boldsymbol{S}=\boldsymbol{E}_1,$$

其中 $\boldsymbol{Q}_1\in\mathbb{R}^{n\times r}$ 满足 $\boldsymbol{Q}_1^{\text{T}}\boldsymbol{Q}_1=\boldsymbol{I}_r$. 则存在 $\mu_1,\cdots,\mu_r\in\lambda(\boldsymbol{A})$ 使得

$$|\mu_k-\lambda_k(\boldsymbol{S})|\leqslant\sqrt{2}\|\boldsymbol{E}_1\|_2,\quad k=1:r.$$

**证明** 令 $\boldsymbol{Q}_2\in\mathbb{R}^{n\times(n-r)}$ 任意矩阵, 使得 $\boldsymbol{Q}=[\boldsymbol{Q}_1,\boldsymbol{Q}_2]$ 是正交矩阵. 可以推出

$$\boldsymbol{Q}^{\text{T}}\boldsymbol{A}\boldsymbol{Q}=\begin{bmatrix}\boldsymbol{S} & 0\\ 0 & \boldsymbol{Q}_2^{\text{T}}\boldsymbol{A}\boldsymbol{Q}_2\end{bmatrix}+\begin{bmatrix}\boldsymbol{Q}_1^{\text{T}}\boldsymbol{E}_1 & \boldsymbol{E}_1^{\text{T}}\boldsymbol{Q}_2\\ \boldsymbol{Q}_2^{\text{T}}\boldsymbol{E}_1 & 0\end{bmatrix}\equiv\boldsymbol{B}+\boldsymbol{E}.$$

于是应用推论 8.1.6 可知, 对 $k=1:n$ 都有 $|\lambda_k(\boldsymbol{A})-\lambda_k(\boldsymbol{B})|\leqslant\|\boldsymbol{E}\|_2$. 由于 $\lambda(\boldsymbol{S})\subseteq\lambda(\boldsymbol{B})$, 存在 $\mu_1,\cdots,\mu_r\in\lambda(\boldsymbol{A})$ 使得

$$|\mu_k-\lambda_k(\boldsymbol{S})|\leqslant\|\boldsymbol{E}\|_2,\quad k=1:r.$$

注意到对任意 $\boldsymbol{x}\in\mathbb{R}^r$ 和 $\boldsymbol{y}\in\mathbb{R}^{n-r}$, 我们有

$$\left\|\boldsymbol{E}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix}\right\|_2\leqslant\|\boldsymbol{E}_1\boldsymbol{x}\|_2+\|\boldsymbol{E}_1^{\text{T}}\boldsymbol{Q}_2\boldsymbol{y}\|_2\leqslant\|\boldsymbol{E}_1\|_2\|\boldsymbol{x}\|_2+\|\boldsymbol{E}_1\|_2\|\boldsymbol{y}\|_2,$$

易知 $\|\boldsymbol{E}\|_2 \leqslant \sqrt{2}\|\boldsymbol{E}_1\|_2$, 从而定理得证. □

**例 8.1.7** 如果

$$\boldsymbol{A} = \begin{bmatrix} 6.8 & 2.4 \\ 2.4 & 8.2 \end{bmatrix}, \quad \boldsymbol{Q}_1 = \begin{bmatrix} 0.7994 \\ 0.6007 \end{bmatrix}, \boldsymbol{S} = (5.1) \in \mathbb{R},$$

则

$$\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{S} = \begin{bmatrix} -0.0828 \\ -0.0562 \end{bmatrix} = \boldsymbol{E}_1.$$

定理预示着 $\boldsymbol{A}$ 在以 5.1 为中心、$\sqrt{2}\|\boldsymbol{E}_1\|_2 \approx 0.1415$ 为半径的邻域内有一个特征值. 由 $\lambda(\boldsymbol{A}) = \{5, 10\}$ 可知该结论正确.

定理 8.1.13 的特征值范围依赖 $\|\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{S}\|_2$. 给定 $\boldsymbol{A}$ 和 $\boldsymbol{Q}_1$, 下列定理表明怎样选择 $\boldsymbol{S}$ 使该量在 Frobenius 范数意义下极小化.

**定理 8.1.14** 如果 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是对称矩阵且 $\boldsymbol{Q}_1 \in \mathbb{R}^{n\times r}$ 的列向量正交. 则

$$\min_{\boldsymbol{S}\in\mathbb{R}^{r\times r}} \|\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{S}\|_F = \|(\boldsymbol{I} - \boldsymbol{Q}_1\boldsymbol{Q}_1^{\mathrm{T}})\boldsymbol{A}\boldsymbol{Q}_1\|_F,$$

且 $\boldsymbol{S} = \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1$ 是极小化矩阵.

**证明** 令 $\boldsymbol{Q}_2 \in \mathbb{R}^{n\times(n-r)}$ 使得 $\boldsymbol{Q} = [\boldsymbol{Q}_1 \quad \boldsymbol{Q}_2]$ 为正交矩阵, 对任意 $\boldsymbol{S} \in \mathbb{R}^{r\times r}$ 我们有

$$\begin{aligned}\|\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1\boldsymbol{S}\|_F^2 &= \|\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{S}\|_F^2 \\ &= \|\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{S}\|_F^2 + \|\boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1\|_F^2.\end{aligned}$$

很清楚, $\boldsymbol{S} = \boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1$ 使上式达到极小. □

这个结论使我们能从任何一个 $r$ 维子空间 $\mathrm{ran}(Q_1)$ 中选取 $r$ 个"最佳的"近似于特征值–特征向量的集合.

**定理 8.1.15** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵且 $\boldsymbol{Q}_1 \in \mathbb{R}^{n\times r}$ 满足 $\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{Q}_1 = \boldsymbol{I}_r$. 若

$$\boldsymbol{Z}^{\mathrm{T}}(\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1)\boldsymbol{Z} = \mathrm{diag}(\theta_1, \cdots, \theta_r) = \boldsymbol{D}$$

是 $\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1$ 的 Schur 分解且 $\boldsymbol{Q}_1\boldsymbol{Z} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_r]$, 则对 $k = 1:r$ 有

$$\|\boldsymbol{A}\boldsymbol{y}_k - \theta_k\boldsymbol{y}_k\|_2 = \|(\boldsymbol{I} - \boldsymbol{Q}_1\boldsymbol{Q}_1^{\mathrm{T}})\boldsymbol{A}\boldsymbol{Q}_1\boldsymbol{Z}\boldsymbol{e}_k\|_2 \leqslant \|(\boldsymbol{I} - \boldsymbol{Q}_1\boldsymbol{Q}_1^{\mathrm{T}})\boldsymbol{A}\boldsymbol{Q}_1\|_2.$$

**证明**

$$\boldsymbol{A}\boldsymbol{y}_k - \theta_k\boldsymbol{y}_k = \boldsymbol{A}\boldsymbol{Q}_1\boldsymbol{Z}\boldsymbol{e}_k - \boldsymbol{Q}_1\boldsymbol{Z}\boldsymbol{D}\boldsymbol{e}_k = (\boldsymbol{A}\boldsymbol{Q}_1 - \boldsymbol{Q}_1(\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_1))\boldsymbol{Z}\boldsymbol{e}_k$$

两边取范数, 定理得证. □

在定理 8.1.15 中, $\theta_k$ 称为 Ritz值, $\boldsymbol{y}_k$ 称为 Ritz向量, $(\theta_k, \boldsymbol{y}_k)$ 称为 Ritz对.

如果我们弱化定理中 $\boldsymbol{Q}_1$ 的列向量为正交的假设, 则定理 8.1.13 的应用更广. 如同所料, 没有正交性其界会变坏.

**定理 8.1.16** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵且

$$\boldsymbol{A}\boldsymbol{X}_1 - \boldsymbol{X}_1\boldsymbol{S} = \boldsymbol{F}_1,$$

其中 $\boldsymbol{X}_1 \in \mathbb{R}^{n\times r}$ 且 $\boldsymbol{S} = \boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}_1$. 如果

$$\|\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{X}_1 - \boldsymbol{I}_r\|_2 = \tau < 1, \tag{8.1.4}$$

*则存在 $\mu_1,\cdots,\mu_r\in\lambda(\boldsymbol{A})$ 使得对 $k=1{:}r$ 都有*

$$|\mu_k-\lambda_k(\boldsymbol{S})|\leqslant\sqrt{2}(\|\boldsymbol{F}_1\|_2+\tau(2+\tau)\|\boldsymbol{A}\|_2).$$

**证明** 令 $\boldsymbol{X}_1=\boldsymbol{ZP}$ 是 $\boldsymbol{X}_1$ 的极分解. 回忆 4.2.10 节中的定义, 这意味着 $\boldsymbol{Z}\in\mathbb{R}^{n\times r}$ 的列向量正交且 $\boldsymbol{P}\in\mathbb{R}^{k\times k}$ 是对称半正定矩阵, 满足 $\boldsymbol{P}^2=\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{X}_1$. 在等式

$$\begin{aligned}\boldsymbol{E}_1\equiv\boldsymbol{AZ}-\boldsymbol{ZS}&=(\boldsymbol{AX}_1-\boldsymbol{X}_1\boldsymbol{S})+\boldsymbol{A}(\boldsymbol{Z}-\boldsymbol{X}_1)-(\boldsymbol{Z}-\boldsymbol{X}_1)\boldsymbol{S}\\&=\boldsymbol{F}_1+\boldsymbol{AZ}(\boldsymbol{I}-\boldsymbol{P})-\boldsymbol{Z}(\boldsymbol{I}-\boldsymbol{P})\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{AX}_1\end{aligned}$$

两边取范数得

$$\|\boldsymbol{E}_1\|_2\leqslant\|\boldsymbol{F}_1\|_2+\|\boldsymbol{A}\|_2\|\boldsymbol{I}-\boldsymbol{P}\|_2(1+\|\boldsymbol{X}_1\|_2^2).\tag{8.1.5}$$

等式 (8.1.4) 意味着

$$\|\boldsymbol{X}_1\|_2^2\leqslant1+\tau.\tag{8.1.6}$$

由于 $\boldsymbol{P}$ 是半正定矩阵, $(\boldsymbol{I}+\boldsymbol{P})$ 非奇异且

$$\boldsymbol{I}-\boldsymbol{P}=(\boldsymbol{I}+\boldsymbol{P})^{-1}(\boldsymbol{I}-\boldsymbol{P}^2)=(\boldsymbol{I}+\boldsymbol{P})^{-1}(\boldsymbol{I}-\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{X}_1),$$

这隐含着 $\|\boldsymbol{I}-\boldsymbol{P}\|_2\leqslant\tau$. 将此不等式和 (8.1.6) 代入 (8.1.5), 有 $\|\boldsymbol{E}_1\|_2\leqslant\|\boldsymbol{F}_1\|_2+\tau(2+\tau)\|\boldsymbol{A}\|_2$. 注意到, 我们可应用定理 8.1.13($\boldsymbol{Q}_1=\boldsymbol{Z}$) 并通过余量阵 $\boldsymbol{E}_1$ 将 $\boldsymbol{A}$ 和 $\boldsymbol{S}$ 的特征值联系起来. □

### 8.1.5 惯性定律

对称矩阵 $\boldsymbol{A}$ 的惯性是三个非负整数的数组 $(m,z,p)$, 其中 $m,z$ 和 $p$ 分别是 $\lambda(\boldsymbol{A})$ 的负元素, 零元素及正元素的个数.

**定理 8.1.17 (Sylvester 惯性定律)** *如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 是对称矩阵且 $\boldsymbol{X}\in\mathbb{R}^{n\times n}$ 非奇异, 则 $\boldsymbol{A}$ 和 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX}$ 有相同的惯性.*

**证明** 设对某 $r$ 有 $\lambda_r(\boldsymbol{A})>0$, 且定义 $S_0\subseteq\mathbb{R}^n$ 为

$$S_0=\mathrm{span}\{\boldsymbol{X}^{-1}\boldsymbol{q}_1,\cdots,\boldsymbol{X}^{-1}\boldsymbol{q}_r\},\quad \boldsymbol{q}_i\neq\boldsymbol{0}$$

其中 $\boldsymbol{Aq}_i=\lambda_i(\boldsymbol{A})\boldsymbol{q}_i,i=1:r$. 从 $\lambda_r(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})$ 的极小化极大特性得

$$\lambda_r(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})=\max_{\dim(S)=r}\min_{\boldsymbol{y}\in S}\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}\geqslant\min_{\boldsymbol{y}\in S_0}\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}.$$

因为

$$\begin{aligned}\boldsymbol{y}\in\mathbb{R}^n&\Rightarrow\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\boldsymbol{y}}{y^{\mathrm{T}}y}\geqslant\sigma_n(\boldsymbol{X})^2,\\\boldsymbol{y}\in S_0&\Rightarrow\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}\geqslant\lambda_r(\boldsymbol{A}),\end{aligned}$$

所以

$$\lambda_r(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})\geqslant\min_{y\in S_0}\left\{\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{AX})\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\boldsymbol{y}}\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}\right\}\geqslant\lambda_r(\boldsymbol{A})\sigma_n(\boldsymbol{X})^2.$$

交换 $A$ 与 $X^TAX$ 的位置, 可类似推出

$$\lambda_r(A) \geqslant \lambda_r(X^TAX)\sigma_n(X^{-1})^2 = \frac{\lambda_r(X^TAX)}{\sigma_1(X)^2}.$$

这表明 $\lambda_r(A)$ 和 $\lambda_r(X^TAX)$ 有相同符号, 这样我们知道 $A$ 和 $X^TAX$ 有相同个数的正特征值. 若将此结果用到 $-A$, 就得到 $A$ 和 $X^TAX$ 有相同个数的负特征值. 显然两个矩阵的零特征值个数也一样. □

**例 8.1.8** 若 $A$=diag(3, 2, −1) 和

$$X = \begin{bmatrix} 1 & 4 & 5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix},$$

则

$$X^TAX = \begin{bmatrix} 3 & 12 & 15 \\ 12 & 50 & 64 \\ 15 & 64 & 82 \end{bmatrix}$$

以及 $\lambda(X^TAX) = [134.769 \quad 0.3555 \quad -0.1252]$.

## 习 题

**8.1.1** 不用本节任何结论, 证明 $2\times 2$ 对称矩阵的特征值一定为实数.

**8.1.2** 计算 $A = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}$ 的 Schur 分解.

**8.1.3** 证明 Hermit 矩阵 ($A^H = A$) 的特征值为实数, 对本节的每一定理及推论, 陈述和证明关于 Hermit 矩阵的相应结论, 当 $A$ 是反对称矩阵时, 有什么类似结论? (提示: 若 $A^T = -A$, 则 i$A$ 为 Hermit 型).

**8.1.4** 证明若 $X \in \mathbb{R}^{n\times r}, r \leqslant n$, 且 $\|X^TX - I\| = r < 1$, 则 $\sigma_{\min}(X) \geqslant 1 - \tau$.

**8.1.5** 设 $A, E \in \mathbb{R}^{n\times n}$ 是对称矩阵并考虑 Schur 分解 $A + tE = QDQ^T$, 其中我们设 $Q = Q(t)$ 和 $D = D(t)$ 是关于 $t \in \mathbb{R}$ 的连续可微函数. 证明: $\dot{D}(t) = \text{diag}(Q(t)^TEQ(t))$, 其中右端矩阵是 $Q(t)^TEQ(t)$ 的对角部分. 通过在等式两边同时从 0 到 1 积分并取 Frobenius 范数, 即可证明 Wielandt-Hoffman 定理.

$$\|D(1) - D(0)\|_F \leqslant \int_0^1 \|\text{diag}(Q(t)^TEQ(t))\|_F \text{d}t \leqslant \|E\|_F.$$

**8.1.6** 证明定理 8.1.5.

**8.1.7** 证明定理 8.1.7.

**8.1.8** 若 $C \in \mathbb{R}^{n\times n}$ 则迹函数$\text{tr}(C) = c_{11} + \cdots + c_{nn}$ 等于 $C$ 的特征值之和. 用此结论证明定理 8.1.8.

**8.1.9** 证明若 $B \in \mathbb{R}^{m\times m}$ 和 $C \in \mathbb{R}^{n\times n}$ 为对称矩阵, 则 $\text{sep}(B,C) = \min\|BX - XC\|_F$, 其中极小值是在所有 $\mathbb{R}^{m\times n}$ 中取.

**8.1.10** 证明不等式 (8.1.3).

**8.1.11** 设 $A \in \mathbb{R}^{n\times n}$ 为对称矩阵, $C \in \mathbb{R}^{n\times r}$ 列满秩且设 $r \ll n$. 应用定理 8.1.8, 找到 $A + CC^T$ 的特征值和 $A$ 的特征值之间的关系.

### 本节注释与参考文献

对称特征值问题的扰动理论的全面综述可见 Wilkinson(1965, 第 2 章), Parlett(1980, 第 10 章和第 11 章), Stewart and Sun(1990, 第 4 章和第 5 章). 在这个研究成熟的领域里的一些代表性的论文包括:

G. W. Stewart (1973). "Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems," *SIAM Review 15,* 727–764.

C. C. Paige (1974). "Eigenvalues of Perturbed Hermitian Matrices," *Lin. Alg. and Its Applic.* 8, 1–10.

A. Ruhe (1975). "On the Closeness of Eigenvalues and Singular Values for Almost Normal Matrices," *Lin. Alg. and Its Applic.* 11, 87–94.

W. Kahan (1975). "Spectra of Nearly Hermitian Martrices," *Proc. Amer. Math. Soc.* 48, 11–17.

A. Schonhage (1979). "Arbitrary Perturbations of Hermitian Matrices," *Lin. Alg. and Its Applic.* 24,143–149.

P. Deift, T. Nanda, and C. Tomei (1983). "Ordinary Differential Equations and the Symmetric Eigenvalue Problem," *SIAMJ. Numer. Anal.* 20, 1–22.

D. S. Scott (1985). "On the Accuracy of the Gershgorin Circle Theorem for Bounding the Spread of a Real Symmetric Matrix," *Lin. Alg. and Its Applic. 65,* 147–155.

J.-G Sun (1995). "A Note on Backward Error Perturbations for the Hermitian Eigenvalue Problem," *BIT* 35, 385–393.

R.-C Li (1996). "Relative Perturbation Theory (I) Eigenvalue and Singular Value Variations," Technical Report UCB//CSD-94-855, Department of EECS, University of California at Berkeley.

R. -C Li (1996). "Relative Perturbation Theory (II) Eigenspace and Singular Subspace Variations," Technical Report UCB//CSD-94-856, Department of EECS, University of California at Berkeley.

## 8.2 幂迭代法

设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵且 $\boldsymbol{U}_0 \in \mathbb{R}^{n\times n}$ 为正交矩阵. 考虑下列 QR迭代:

$$
\begin{aligned}
&T_0 = U_0^{\mathrm{T}} A U_0 \\
&\textbf{for } k = 1, 2, \cdots \\
&\qquad T_{k-1} = U_k R_k \qquad \text{(QR分解)} \\
&\qquad T_k = R_k U_k \\
&\textbf{end}
\end{aligned}
\tag{8.2.1}
$$

由于 $\boldsymbol{T}_k = \boldsymbol{R}_k\boldsymbol{U}_k = \boldsymbol{U}_k^{\mathrm{T}}(\boldsymbol{U}_k\boldsymbol{R}_k)\boldsymbol{U}_k = \boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{T}_{k-1}\boldsymbol{U}_k$, 由归纳法知

$$
\boldsymbol{T}_k = (\boldsymbol{U}_0\boldsymbol{U}_1\cdots\boldsymbol{U}_k)^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{U}_0\boldsymbol{U}_1\cdots\boldsymbol{U}_k). \tag{8.2.2}
$$

这样, 每个 $\boldsymbol{T}_k$ 正交相似于 $\boldsymbol{A}$. 而且, $\boldsymbol{T}_k$ 几乎总是收敛到对角型. 所以, 可以说 (8.2.1) 几乎总是“收敛”到 $\boldsymbol{A}$ 的 Schur 分解. 为了建立这个重要结论, 我们首先考虑幂迭代法和正交迭代法.

### 8.2.1 幂法

给定一个 2 范数单位向量 $\boldsymbol{q}^{(0)} \in \mathbb{R}^n$, 幂方法产生一列如下向量 $q^{(k)}$:

**for** $k = 1, 2, \cdots$

$$z^{(k)} = Aq^{(k-1)}$$

$$q^{(k)} = z^{(k)}/\|z^{(k)}\|_2 \tag{8.2.3}$$

$$\lambda^{(k)} = [q^{(k)}]^{\mathrm{T}} A q^{(k)}$$

**end**

如果 $\boldsymbol{q}^{(0)}$ 不是“秩亏损”的且 $\boldsymbol{A}$ 的极大模特征值唯一, 则 $\boldsymbol{q}^{(k)}$ 收敛到一特征向量.

**定理 8.2.1** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵且

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \mathrm{diag}(\lambda_1, \cdots, \lambda_n),$$

其中 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n]$ 是正交矩阵且 $|\lambda_1| > |\lambda_2| \geqslant \cdots |\lambda_n|$. 令向量 $\boldsymbol{q}^{(k)}$ 是由 (8.2.3) 得到的且定义 $\theta_k \in \left[0, \dfrac{\pi}{2}\right]$ 如下:

$$\cos(\theta_k) = |\boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}^{(k)}|,$$

若 $\cos(\theta_0) \neq 0$, 则

$$|\sin(\theta_k)| \leqslant \tan(\theta_0)\left|\frac{\lambda_2}{\lambda_1}\right|^k, \tag{8.2.4}$$

$$|\lambda^{(k)} - \lambda_1| \leqslant |\lambda_1 - \lambda_n|\tan(\theta_0)^2\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}. \tag{8.2.5}$$

**证明** 由迭代定义, 可知 $\boldsymbol{q}^{(k)}$ 是 $\boldsymbol{A}^k\boldsymbol{q}^{(0)}$ 的倍数且

$$|\sin(\theta_k)|^2 = 1 - (\boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}^{(k)})^2 = 1 - \left(\frac{\boldsymbol{q}_1^{T}\boldsymbol{A}^k\boldsymbol{q}^{(0)}}{\|\boldsymbol{A}^k\boldsymbol{q}^{(0)}\|_2}\right)^2.$$

如果 $\boldsymbol{q}^{(0)}$ 有特征向量展式 $\boldsymbol{q}^{(0)} = a_1\boldsymbol{q}_1 + \cdots + a_n\boldsymbol{q}_n$, 则

$$|a_1| = |\boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{q}^{(0)}| = \cos(\theta_0) \neq 0,$$

$$a_1^2 + \cdots + a_n^2 = 1,$$

且 $\boldsymbol{A}^k\boldsymbol{q}^{(0)} = a_1\lambda_1^k\boldsymbol{q}_1 + a_2\lambda_2^k\boldsymbol{q}_2 + \cdots + a_n\lambda_n^k\boldsymbol{q}_n$. 于是

$$|\sin(\theta_k)|^2 = 1 - \frac{a_1^2\lambda_1^{2k}}{\sum_{i=1}^n a_i^2\lambda_i^{2k}} = \frac{\sum_{i=2}^n a_i^2\lambda_i^{2k}}{\sum_{i=1}^n a_i^2\lambda_i^{2k}}$$

$$
\begin{aligned}
&\leqslant \frac{\sum_{i=2}^{n} a_i^2 \lambda_i^{2k}}{a_1^2 \lambda_1^{2k}} = \frac{1}{a_1^2} \sum_{i=2}^{n} a_i^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k} \\
&\leqslant \frac{1}{a_1^2} \left(\sum_{i=2}^{n} a_i^2\right) \left(\frac{\lambda_2}{\lambda_1}\right)^{2k} = \frac{1-a_1^2}{a_1^2} \left(\frac{\lambda_2}{\lambda_1}\right)^{2k} \\
&= \tan(\theta_0)^2 \left(\frac{\lambda_2}{\lambda_1}\right)^{2k}.
\end{aligned}
$$

这证明了式 (8.2.4). 类似地,

$$
\lambda^{(k)} = [q^{(k)}]^{\mathrm{T}} A q^{(k)} = \frac{[q^{(0)}]^{\mathrm{T}} A^{2k+1} q^{(0)}}{[q^{(0)}]^{\mathrm{T}} A^{2k} q^{(0)}} = \frac{\sum_{i=1}^{n} a_i^2 \lambda_i^{2k+1}}{\sum_{i=1}^{n} a_i^2 \lambda_i^{2k}}.
$$

于是

$$
\begin{aligned}
|\lambda^{(k)} - \lambda_1| &= \left| \frac{\sum_{i=2}^{n} a_i^2 \lambda_i^{2k} (\lambda_i - \lambda_1)}{\sum_{i=1}^{n} a_i^2 \lambda_i^{2k}} \right| \leqslant |\lambda_1 - \lambda_n| \frac{1}{a_1^2} \sum_{i=2}^{n} a_i^2 \left(\frac{\lambda_i}{\lambda_1}\right)^{2k} \\
&\leqslant |\lambda_1 - \lambda_n| \tan(\theta_0)^2 \left(\frac{\lambda_2}{\lambda_1}\right)^{2k}.
\end{aligned}
$$

□

**例 8.2.1**

$$
A = \begin{bmatrix} -1.6407 & 1.0814 & 1.2014 & 1.1539 \\ 1.0814 & 4.1573 & 7.4035 & -1.0463 \\ 1.2014 & 7.4035 & 2.7890 & -1.5737 \\ 1.1539 & -1.0463 & -1.5737 & 8.6944 \end{bmatrix}
$$

的特征值由 $\lambda(A) = \{12, 8, -4, -2\}$ 给出. 若 (8.2.3) 应用到此矩阵, 令 $q^{(0)} = [1\ \ 0\ \ 0\ \ 0]^{\mathrm{T}}$, 则

| $k$ | $\lambda^{(k)}$ | $k$ | $\lambda^{(k)}$ |
|---|---|---|---|
| 1 | 2.3156 | 6 | 11.7747 |
| 2 | 8.6802 | 7 | 11.8967 |
| 3 | 10.3163 | 8 | 11.9534 |
| 4 | 11.0663 | 9 | 11.9792 |
| 5 | 11.5259 | 10 | 11.9907 |

注意到以速度 $|\lambda_2/\lambda_1|^{2k} = (8/12)^{2k} = (4/9)^k$ 收敛到 $\lambda_1 = 12$.

利用定理 8.1.13 可得到幂法的计算误差界. 如果

$$
\|Aq^{(k)} - \lambda^{(k)} q^{(k)}\|_2 = \delta,
$$

则存在 $\lambda \in \lambda(A)$ 使得 $|\lambda^{(k)} - \lambda| \leqslant \sqrt{2}\delta$.

### 8.2.2　逆迭代

假设幂法中的 $A$ 由 $(A - \lambda I)^{-1}$ 代替. 如果 $\lambda$ 很接近 $A$ 的一个单根, 则下一

个迭代向量在相应的特征方向的成分就非常多.

$$\left.\begin{array}{l} \boldsymbol{x} = \sum_{i=1}^{n} a_i \boldsymbol{q}_i \\ \boldsymbol{A}\boldsymbol{q}_i = \lambda_i \boldsymbol{q}_i, i = 1:n \end{array}\right\} \Rightarrow (\boldsymbol{A} - \lambda \boldsymbol{I})^{-1}\boldsymbol{x} = \sum_{i=1}^{n} \frac{a_i}{\lambda_i - \lambda}\boldsymbol{q}_i.$$

这样, 若 $\lambda \approx \lambda_j$ 且 $a_j$ 不是太小, 则在 $\boldsymbol{q}_j$ 方向上此向量有一较强分量. 这个过程称为逆迭代且需要求解以 $\boldsymbol{A} - \lambda\boldsymbol{I}$ 为系数矩阵的线性方程组.

### 8.2.3 Rayleigh 商迭代

设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是对称矩阵且 $x$ 是一给定的非零 $n$ 维向量. 简单的求导数就发现

$$\lambda = r(\boldsymbol{x}) \equiv \frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}$$

使 $\|(\boldsymbol{A} - \lambda\boldsymbol{I})\boldsymbol{x}\|_2$ 达到极小 (参看定理 8.1.14). 标量 $r(x)$ 称为 $\boldsymbol{x}$ 的 Rayleigh商. 显然, 若 $\boldsymbol{x}$ 是近似的特征向量, 则 $r(x)$ 是其对应特征值的较好估计. 把这个思想与逆迭代结合就得到 Rayleigh商迭代:

给定 $x_0, \|x_0\|_2 = 1$

**for** $k = 0, 1, \cdots$

  $\mu_k = r(x_k)$ (8.2.6)

  解$(A - \mu_k I)z_{k+1} = x_k$, 得$z_{k+1}$

  $x_{k+1} = z_{k+1}/\|z_{k+1}\|_2$

**end**

**例 8.2.2** 若把 (8.2.6) 应用到

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 3 & 6 & 10 & 15 & 21 \\ 1 & 4 & 10 & 20 & 35 & 56 \\ 1 & 5 & 15 & 35 & 70 & 126 \\ 1 & 6 & 21 & 56 & 126 & 252 \end{bmatrix},$$

其中 $\boldsymbol{x}_0 = [1,1,1,1,1,1]^{\mathrm{T}}/6$, 则

| $k$ | $\mu_k$ | $k$ | $\mu_k$ |
|---|---|---|---|
| 0 | 153.8333 | 3 | 13.8687 |
| 1 | 120.0571 | 4 | 15.4959 |
| 2 | 49.5011 | 5 | 15.5534 |

迭代收敛到特征值 $\lambda = 15.553\,473\,273\,7$.

Rayleigh 商迭代几乎总是收敛的, 当收敛时, 其收敛速度为三次. 我们演示 $n = 2$ 的情形. 不失一般性, 可假定 $\boldsymbol{A} = \mathrm{diag}(\lambda_1, \lambda_2)$, 其中 $\lambda_1 > \lambda_2$. 记 $\boldsymbol{x}_k$ 为

$$\boldsymbol{x}_k = \begin{bmatrix} c_k \\ s_k \end{bmatrix}, \quad c_k^2 + s_k^2 = 1,$$

可以得出 (8.2.6) 中的 $\mu_k = \lambda_1 c_k^2 + \lambda_1 s_k^2$, 且

$$z_{k+1} = \frac{1}{\lambda_1 - \lambda_2}\begin{bmatrix} c_k/s_k^2 \\ -s_k/c_k^2 \end{bmatrix}.$$

计算表明

$$c_{k+1} = \frac{c_k^3}{\sqrt{c_k^6 + s_k^6}}, \quad s_{k+1} = \frac{-s_k^3}{\sqrt{c_k^6 + s_k^6}}. \tag{8.2.7}$$

从这些等式很清楚看到, 若 $|c_k| \neq |s_k|, x_k$ 立方收敛到 $\text{span}\{e_1\}$ 或 $\text{span}\{e_2\}$.

在 Parlett(1974) 中可以找到 Rayleigh 商迭代具体实现的细节.

### 8.2.4　正交迭代

幂法的直接推广可用于计算高维不变子空间. 令 $r$ 是选定的满足 $1 \leqslant r \leqslant n$ 的整数. 给定一个 $n \times r$ 的列正交矩阵 $\boldsymbol{Q}_0$, 正交迭代方法产生如下一列矩阵 $\{\boldsymbol{Q}_k\} \subseteq \mathbb{R}^{n\times r}$:

$$\begin{aligned} &\textbf{for } k = 1, 2, \cdots \\ &\quad Z_k = AQ_{k-1} \\ &\quad Q_k R_k = Z_k \quad (\text{QR分解}) \\ &\textbf{end} \end{aligned} \tag{8.2.8}$$

注意到如果 $r = 1$, 这正是幂法. 而且,$\{\boldsymbol{Q}_k e_1\}$ 恰是取初值 $q(0) = \boldsymbol{Q}_0 e_1$ 用幂法产生的向量序列.

为了分析 (8.2.8) 的表现, 设

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{D} = \text{diag}(\lambda_i), \quad |\lambda_1| \geqslant |\lambda_2| \geqslant \cdots \geqslant |\lambda_n| \tag{8.2.9}$$

是 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 的 Schur 分解, 将 $\boldsymbol{Q}, \boldsymbol{D}$ 划分如下:

$$\boldsymbol{Q} = [\underset{r}{\boldsymbol{Q}_\alpha} \quad \underset{n-r}{\boldsymbol{Q}_\beta}] \qquad \boldsymbol{D} = \begin{bmatrix} \boldsymbol{D}_1 & 0 \\ 0 & \boldsymbol{D}_2 \end{bmatrix}\begin{matrix} r \\ n-r \end{matrix} \tag{8.2.10}$$
$$\qquad\qquad\qquad\qquad\qquad\qquad r \qquad n-r$$

若 $|\lambda_r| > |\lambda_{r+1}|$, 则

$$D_r(\boldsymbol{A}) = \text{ran}(\boldsymbol{Q}_\alpha)$$

是 $r$ 维的主不变子空间. 这是与特征值 $\lambda_1, \cdots, \lambda_r$ 相对应的唯一不变子空间.

下面的定理表明了在合理假设下, 由 (8.2.8) 产生的子空间 $\text{ran}(\boldsymbol{Q}_k)$ 以速度 $|\lambda_{r+1}/\lambda_r|^k$ 收敛到 $D_r(\boldsymbol{A})$.

**定理 8.2.2**　令 $\boldsymbol{A} \in \mathbb{R}^{n\times n} (n \geqslant 2)$ 的 Schur 分解由 (8.2.9) 和 (8.2.10) 给出. 假定 $|\lambda_r| > |\lambda_{r+1}|$ 且 $n \times r$ 矩阵 $\{\boldsymbol{Q}_k\}$ 由 (8.2.8) 所定义. 如果 $\theta \in [0, \pi/2]$ 满足

$$\cos(\theta) = \min_{\substack{u \in D_r(\boldsymbol{A}) \\ v \in \text{ran}(\boldsymbol{Q}_0)}} \frac{|u^{\mathrm{T}} v|}{\|u\|_2 \|v\|_2} > 0,$$

则 $\text{dist}(D_r(\boldsymbol{A}), \text{ran}(\boldsymbol{Q}_k)) \leqslant \tan(\theta) \left|\dfrac{\lambda_{r+1}}{\lambda_r}\right|^k$. 参见定理 7.3.1.

**证明** 由归纳法可证

$$A^kQ_0 = Q_k(R_k\cdots R_1),$$

且由 (8.2.10) 的划分我们有

$$\begin{bmatrix} D_1^k & 0 \\ 0 & D_2^k \end{bmatrix}\begin{bmatrix} Q_\alpha^{\mathrm{T}}Q_0 \\ Q_\beta^{\mathrm{T}}Q_0 \end{bmatrix} = \begin{bmatrix} Q_\alpha^{\mathrm{T}}Q_k \\ Q_\beta^{\mathrm{T}}Q_k \end{bmatrix}(R_k\cdots R_1).$$

如果 $Q^{\mathrm{T}}Q_k = [Q_\alpha, Q_\beta]^{\mathrm{T}}Q_k = \begin{bmatrix} Q_\alpha^{\mathrm{T}}Q_k \\ Q_\beta^{\mathrm{T}}Q_k \end{bmatrix} \equiv \begin{bmatrix} V_k \\ W_k \end{bmatrix}$, 则

$$\begin{aligned}
\cos(\theta_{\min}) = \sigma_r(V_0) &= \sqrt{1-\|W_0\|_2^2},\\
\mathrm{dist}(D_r(A), \mathrm{ran}(Q_k)) &= \|W_k\|_2,\\
D_1^kV_0 &= V_k(R_k\cdots R_1),\\
D_2^kW_0 &= W_k(R_k\cdots R_1).
\end{aligned}$$

由上可知 $V_0$ 非奇异, 这说明 $V_k$ 和 $(R_k\cdots R_1)$ 也非奇异. 这样

$$\begin{aligned}
W_k = D_2^kW_0(R_k\cdots R_1)^{-1} &= D_2^kW_0(V_k^{-1}D_1^kV_0)^{-1}\\
&= D_2^kW_0V_0^{-1}D_1^{-k}V_k,
\end{aligned}$$

且

$$\begin{aligned}
\|W_k\|_2 &\leqslant \|D_2^k\|_2\|W_0\|_2\|V_0^{-1}\|_2\|D_1^{-k}\|_2\|V_k\|_2\\
&\leqslant |\lambda_{r+1}|^k\sin(\theta)\frac{1}{\cos(\theta)}\frac{1}{|\lambda_r|^k} = \tan(\theta)\left|\frac{\lambda_{r+1}}{\lambda_r}\right|^k.
\end{aligned}$$ □

**例 8.2.3** 若把 (8.2.8) 应用到例 8.2.1 的矩阵, 取 $r=2$ 且 $Q_0 = I_4(:,1:2)$, 则

| $k$ | $\mathrm{dist}(D_2(A), \mathrm{ran}(Q_k))$ | $k$ | $\mathrm{dist}(D_2(A), \mathrm{ran}(Q_k))$ |
|---|---|---|---|
| 1 | 0.8806 | 6 | 0.0044 |
| 2 | 0.4091 | 7 | 0.0020 |
| 3 | 0.1121 | 8 | 0.0010 |
| 4 | 0.0313 | 9 | 0.0005 |
| 5 | 0.0106 | 10 | 0.0002 |

### 8.2.5 QR 迭代

考虑当 $r=n$ 时应用正交迭代法 (8.2.8) 的情况. 令 $Q^{\mathrm{T}}AQ = \mathrm{diag}(\lambda_1,\cdots,\lambda_n)$ 是 Schur 分解, 而且设

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|.$$

如果 $Q = [q_1,\cdots,q_n]$, $Q_k = [q_1^{(k)},\cdots,q_n^{(k)}]$ 且对 $i=1:n-1$ 有

$$\mathrm{dist}(D_i(A), \mathrm{span}\{q_1^{(0)},\cdots,q_i^{(0)}\}) < 1. \tag{8.2.11}$$

则由定理 8.2.2 可知, 对 $i=1:n-1$,

$$\mathrm{dist}(\mathrm{span}\{q_1^{(k)},\cdots,q_i^{(k)}\}, \mathrm{span}\{q_1,\cdots,q_i\}) = 0\left(\left|\frac{\lambda_{i+1}}{\lambda_i}\right|^k\right).$$

这说明由

$$T_k = Q_k^{\mathrm{T}} A Q_k$$

定义的矩阵 $T_k$ 收敛到对角型. 这样, 可以说当 $r = n$ 且初始迭代矩阵 $Q_0 \in \mathbb{R}^{n\times n}$ 在 (8.2.11) 意义下非秩亏时, 正交迭代方法计算 Schur 分解.

考虑怎样直接从 $T_{k-1}$ 计算后一个矩阵 $T_k$ 就得到了 QR 迭代法. 一方面, 由 (8.2.1) 和 $T_{k-1}$ 的定义得到

$$T_{k-1} = Q_{k-1}^{\mathrm{T}} A Q_{k-1} = Q_{k-1}^{\mathrm{T}}(AQ_{k-1}) = (Q_{k-1}^{\mathrm{T}} Q_k) R_k.$$

另一方面,

$$T_k = Q_k^{\mathrm{T}} A Q_k = (Q_k^{\mathrm{T}} A Q_{k-1})(Q_{k-1}^{\mathrm{T}} Q_k) = R_k (Q_{k-1}^{\mathrm{T}} Q_k).$$

这样, 计算 $T_{k-1}$ 的 QR 分解, 然后按相反顺序将这些因子乘起来就得到了 $T_k$. 这恰好是 (8.2.1) 所做的事.

**例 8.2.4**　若把 QR 迭代 (8.2.1) 应用到例 8.2.1 的矩阵, 则 10 次迭代后

$$T_{10} = \begin{bmatrix} 11.9907 & -0.1926 & -0.0004 & 0.0000 \\ -0.1926 & 8.0093 & -0.0029 & 0.0001 \\ -0.0004 & -0.0029 & -4.0000 & 0.0007 \\ 0.0000 & 0.0001 & 0.0007 & -2.0000 \end{bmatrix}$$

$T_k$ 的非对角线元素收敛到零的情况如下:

| $k$ | $\lvert T_k(2,1)\rvert$ | $\lvert T_k(3,1)\rvert$ | $\lvert T_k(4,1)\rvert$ | $\lvert T_k(3,2)\rvert$ | $\lvert T_k(4,2)\rvert$ | $\lvert T_k(4,3)\rvert$ |
|---|---|---|---|---|---|---|
| 1 | 3.9254 | 1.8122 | 3.3892 | 4.2492 | 2.8367 | 1.1679 |
| 2 | 2.6491 | 1.2841 | 2.1908 | 1.1587 | 3.1473 | 0.2294 |
| 3 | 2.0147 | 0.6154 | 0.5082 | 0.0997 | 0.9859 | 0.0748 |
| 4 | 1.6930 | 0.2408 | 0.0970 | 0.0723 | 0.2596 | 0.0440 |
| 5 | 1.2928 | 0.0866 | 0.0173 | 0.0665 | 0.0667 | 0.0233 |
| 6 | 0.9222 | 0.0299 | 0.0030 | 0.0405 | 0.0169 | 0.0118 |
| 7 | 0.6346 | 0.0101 | 0.0005 | 0.0219 | 0.0043 | 0.0059 |
| 8 | 0.4292 | 0.0034 | 0.0001 | 0.0113 | 0.0011 | 0.0030 |
| 9 | 0.2880 | 0.0011 | 0.0000 | 0.0057 | 0.0003 | 0.0015 |
| 10 | 0.1926 | 0.0004 | 0.0000 | 0.0029 | 0.0001 | 0.0007 |

注意, 一步 QR 迭代需要 $O(n^3)$ 个 flop. 而且, 由于收敛是线性的 (当收敛存在时), 很明显, 用该方法来计算 Schur 分解的代价是非常昂贵的. 幸运的是, 这些实际困难能够克服, 下一节我们要证明这一点.

## 习　　题

**8.2.1**　设 $A_0 \in \mathbb{R}^{n\times n}$ 为对称正定矩阵, 考虑下列迭代:

**for**　$k = 1, 2, \cdots$

　　$A_{k-1} = G_k G_k^{\mathrm{T}}$　(Cholesky分解)

$A_k = G_k^{\mathrm{T}} G_k$

**end**

(a) 证明上面定义的迭代有意义. (b) 证明若 $\boldsymbol{A}_0 = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$ 有特征值 $\lambda_1 \geqslant \lambda_2 > 0$(其中 $a \geqslant c$), 则 $\boldsymbol{A}_k$ 收敛到对角矩阵 $\mathrm{diag}(\lambda_1, \lambda_2)$.

**8.2.2** 证明 (8.2.7).

**8.2.3** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是对称矩阵且定义函数 $f: \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ 为

$$f\begin{pmatrix} \boldsymbol{x} \\ \lambda \end{pmatrix} = \begin{bmatrix} \boldsymbol{A}\boldsymbol{x} - \lambda \boldsymbol{x} \\ (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x} - 1)/2 \end{bmatrix},$$

其中 $\boldsymbol{x} \in \mathbb{R}^n$ 且 $\lambda \in \mathbb{R}$. 设 $\boldsymbol{x}_+$ 和 $\lambda_+$ 是应用牛顿法于 $f$ 在 $\boldsymbol{x}_c$ 和 $\lambda_c$ 所定义的"当前点"的值产生的. 试给出 $\boldsymbol{x}_+$ 和 $\lambda_+$ 的表达式, 设 $\|\boldsymbol{x}_c\|_2 = 1$ 且 $\lambda_c = \boldsymbol{x}_c^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x}_c$.

### 本节注释与参考文献

下列参考文献讨论正交迭代方法 (也称之为同步迭代方法):

G. W. Stewart (1969). "Accelerating The Orthogonal Iteration for the Eigenvalues of a Hermitian Matrix," *Numer. Math.* 13, 362–376.

M. Clint and A. Jennings (1970). "The Evaluation of Eigenvalues and Eigenvectors of Real Symmetric Matrices by Simultaneous Iteration," *Comp. J.* 13, 76–80.

H. Rutishauser (1970). "Simultaneous Iteration Method for Symmetric Matrices," *Numer. Math. 16,* 205–223. See also Wilkinson and Reinsch (1971, pp. 284–302).

关于 Rayleigh 商方法的参考文献有:

J. Vandergraft (1971). "Generalized Rayleigh Methods with Applications to Finding Eigenvalues of Large Matrices," *Lin. Alg. and Its Applic.* 4, 353–368.

B. N. Parlett (1974). "The Rayleigh Quotient Iteration and Some Generalizations for Nonnormal Matrices," *Math. Comp.* 28, 679–693.

R. A. Tapia and D. L. Whitley (1988). "The Projected Newton Method Has Order $1+\sqrt{2}$ for the Symmetric Eigenvalue Problem," *SIAMJ. Num. Anal.* 25, 1376–1382.

S. Batterson and J. Smillie (1989). "The Dynamics of Rayleigh Quotient Iteration," *SIAM J. Num. Anal.* 26, 624–636.

C. Beattie and D. W. Fox (1989). "Localization Criteria and Containment for Rayleigh Quotient Iteration," *SIAM J. Matrix Anal. Appl.* 10, 80–93.

P. T. P. Tang (1994). "Dynamic Condition Estimation and Rayleigh-Ritz Approximation," *SIAMJ. Matrix Anal. Appl.* 15, 331–346.

## 8.3 对称 QR 算法

有两种方式可使对称 QR 迭代 (8.2.1) 非常有效. 首先, 我们阐明怎样计算一个正交矩阵 $\boldsymbol{U}$ 使得 $\boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U} = \boldsymbol{T}$ 为三对角矩阵. 有了这一约化, (8.2.1) 产生的迭

代矩阵都是三对角矩阵, 使得每步工作量为 $O(n^2)$ 个 flop. 其次, 应用位移思想, 用此技巧, 约化以立方速度收敛到对角型. 这比在 8.2.5 小节中讨论的使非对角线元素以速度 $|\lambda_{i+1}/\lambda_i|^k$ 化为零的方法要好得多.

### 8.3.1 化为三对角阵

如果 $\boldsymbol{A}$ 对称, 则可以找到一正交矩阵 $\boldsymbol{Q}$ 使得

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T} \tag{8.3.1}$$

为三对角矩阵. 我们称此为*三对角分解*. 作为数据压缩, 此方法朝对角化迈出了很大一步.

我们现在来说明怎样用 Householder 矩阵来计算 (8.3.1). 设 Householder 矩阵 $\boldsymbol{P}_1,\cdots,\boldsymbol{P}_{k-1}$ 已确定, 使得若 $\boldsymbol{A}_{k-1}=(\boldsymbol{P}_1\cdots\boldsymbol{P}_{k-1})^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{P}_1\cdots\boldsymbol{P}_{k-1})$, 则

$$\boldsymbol{A}_{k-1}=\begin{bmatrix}\boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{0}\\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23}\\ \boldsymbol{0} & \boldsymbol{B}_{32} & \boldsymbol{B}_{33}\end{bmatrix}\begin{matrix}k-1\\1\\n-k\end{matrix}$$
$$\qquad\qquad k-1 \quad\ 1 \quad\ n-k$$

的前 $k-1$ 列为三对角矩阵. 如果 $\bar{\boldsymbol{P}}_k$ 是 $n-k$ 阶的 Householder 矩阵, 使得 $\bar{\boldsymbol{P}}_k\boldsymbol{B}_{32}$ 是 $\boldsymbol{I}_{n-k}(:,1)$ 的倍数而且如果 $\boldsymbol{P}_k=\mathrm{diag}(\boldsymbol{I}_k;\bar{\boldsymbol{P}}_k)$, 则

$$\boldsymbol{A}_k=\boldsymbol{P}_k\boldsymbol{A}_{k-1}\boldsymbol{P}_k=\begin{bmatrix}\boldsymbol{B}_{11} & \boldsymbol{B}_{12} & \boldsymbol{0}\\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \boldsymbol{B}_{23}\bar{\boldsymbol{P}}_k\\ \boldsymbol{0} & \bar{\boldsymbol{P}}_k\boldsymbol{B}_{32} & \bar{\boldsymbol{P}}_k\boldsymbol{B}_{33}\bar{\boldsymbol{P}}_k\end{bmatrix}\begin{matrix}k-1\\1\\n-k\end{matrix}$$
$$\qquad\qquad k-1 \quad\ 1 \quad\ n-k$$

的右上角的 $k\times k$ 主子矩阵为三对角矩阵. 显然, 若 $\boldsymbol{U}_0=\boldsymbol{p}_1\cdots\boldsymbol{P}_{n-2}$, 则 $\boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{A}\boldsymbol{U}_0=\boldsymbol{T}$ 为三对角矩阵.

在计算 $\boldsymbol{A}_k$ 中, 很重要的一点是形成矩阵 $\bar{\boldsymbol{P}}_k\boldsymbol{B}_{33}\bar{\boldsymbol{P}}_k$ 时充分利用对称性. 精确地说, 设 $\bar{\boldsymbol{P}}_k$ 有如下形状:

$$\bar{\boldsymbol{P}}_k=\boldsymbol{I}-\beta\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}},\quad \beta=2/\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v},\quad 0\neq\boldsymbol{v}\in\mathbb{R}^{n-k}.$$

注意到若 $\boldsymbol{p}=\beta\boldsymbol{B}_{33}\boldsymbol{v}$ 且 $\boldsymbol{w}=\boldsymbol{p}-(\beta\boldsymbol{p}^{\mathrm{T}}\boldsymbol{v}/2)\boldsymbol{v}$, 则

$$\bar{\boldsymbol{P}}_k\boldsymbol{B}_{33}\bar{\boldsymbol{P}}_k=\boldsymbol{B}_{33}-\boldsymbol{v}\boldsymbol{w}^{\mathrm{T}}-\boldsymbol{w}\boldsymbol{v}^{\mathrm{T}}.$$

由于只需计算此矩阵的上三角部分, 我们看到完成从 $\boldsymbol{A}_{k-1}$ 到 $\boldsymbol{A}_k$ 的变换只需 $4(n-k)^2$ 个 flop.

**算法 8.3.1 (Householder 三对角化)** 给定一对角矩阵 $A\in\mathbb{R}^{n\times n}$, 本算法用 $\boldsymbol{T}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{T}$ 为三对角矩阵, $\boldsymbol{Q}=\boldsymbol{H}_1\cdots\boldsymbol{H}_{n-2}$ 是 Householder 变换的乘积.

```
for k = 1 : n-2
    [v, β] = house(A(k+1 : n, k))
```

$p = \beta A(k+1:n, k+1:n)v$
$w = p - (\beta p^{\mathrm{T}} v/2)v$
$A(k+1,k) = \|A(k+1:n,k)\|_2; A(k,k+1) = A(k+1,k)$
$A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - vw^{\mathrm{T}} - wv^{\mathrm{T}}$

**end**

当计算秩 2 修正充分利用对称性时, 算法需 $4n^3/3$ 个 flop. 矩阵 $\boldsymbol{Q}$ 以分解形式储存在 $\boldsymbol{A}$ 的次对角线下面部分. 如果需要显式 $\boldsymbol{Q}$, 则需额外的 $4n^3/3$ 个 flop.

**例 8.3.1**

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.6 & 0.8 \\ 0 & 0.8 & 0.6 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 1 & 3 & 4 \\ 3 & 2 & 8 \\ 4 & 8 & 3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.6 & 0.8 \\ 0 & 0.8 & -0.6 \end{bmatrix} = \begin{bmatrix} 1 & 5 & 0 \\ 5 & 10.32 & 1.76 \\ 0 & 1.76 & -5.32 \end{bmatrix}.$$

注意, 如果 $\boldsymbol{T}$ 有零次对角元, 则特征问题化为两个小特征问题. 确切地说, 若 $t_{k+1,k} = 0$, 则 $\lambda(\boldsymbol{T}) = \lambda(T(1:k,1:k)) \cup \lambda(T(k+1:n,k+1:n))$. 若 $\boldsymbol{T}$ 没有零次对角元, 则说它是不可约的.

令 $\hat{\boldsymbol{T}}$ 为算法 8.3.1 所得到的 $\boldsymbol{T}$ 之计算值. 可以证明 $\hat{\boldsymbol{T}} = \hat{\boldsymbol{Q}}^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E})\hat{\boldsymbol{Q}}$, 其中 $\hat{\boldsymbol{Q}}$ 是精确的正交矩阵, 而 $\boldsymbol{E}$ 是满足 $\|\boldsymbol{E}\|_F \leqslant c\mathbf{u}\|\boldsymbol{A}\|_F$ 的对称矩阵, 这里 $c$ 是小常数. 见 Wilkinson(1965, 297 页).

### 8.3.2 三对角分解的性质

我们来证明关于三对角分解的两个定理, 它们在之后将起关键的作用. 第一个定理将 (8.3.1) 和某一 Krylov矩阵的 QR 分解联系起来, Krylov 矩阵有形式

$$K(\boldsymbol{A}, \boldsymbol{v}, k) = [\boldsymbol{v}, \boldsymbol{A}\boldsymbol{v}, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{v}], \quad \boldsymbol{A} \in \mathbb{R}^{n\times n}, \quad \boldsymbol{v} \in \mathbb{R}^n.$$

**定理 8.3.1** *若 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T}$ 为对称矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 的三对角分解, 则 $\boldsymbol{Q}^{\mathrm{T}}K(A,Q(:,1),n) = \boldsymbol{R}$ 为上三角形矩阵. 若 $\boldsymbol{R}$ 非奇异, 则 $\boldsymbol{T}$ 不可约. 若 $\boldsymbol{R}$ 奇异且 $k$ 是使 $r_{kk} = 0$ 的最小指标, 则 $k$ 也是使 $t_{k,k-1}$ 为零的最小指标. 见定理 7.4.3.*

**证明** 显然, 若 $q_1 = Q(:,1)$, 则

$$\begin{aligned} &\boldsymbol{Q}^{\mathrm{T}}K(\boldsymbol{A},Q(:,1),n) \\ &= [\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{q}_1, (\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q})(\boldsymbol{Q}^{\mathrm{T}}q_1), \cdots, (\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q})^{n-1}(\boldsymbol{Q}^{\mathrm{T}}q_1)] \\ &= [e_1, \boldsymbol{T}e_1, \cdots, \boldsymbol{T}^{n-1}e_1] = \boldsymbol{R} \end{aligned}$$

是上三角形矩阵, 满足 $r_{11} = 1$ 且对 $i = 2:n$ 都有 $r_{ii} = t_{21}t_{32}\cdots t_{i,i-1}$. 显然, 若 $R$ 非奇异, 则 $T$ 不可约. 若 $R$ 奇异且 $r_{kk}$ 是其第一个零对角元, 则 $k \geqslant 2$ 且 $t_{k,k-1}$ 是第一个零次对角元. □

**定理 8.3.2 (隐式Q定理)** *假设 $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n]$ 和 $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n]$ 是正交矩阵使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T}$ 和 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{S}$ 为三对角矩阵, 其中 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 是对称矩阵. 令 $k$ 表示使 $t_{k+1,k} = 0$ 的最小正整数. 约定 $\boldsymbol{T}$ 不可约时 $k = n$. 若 $\boldsymbol{v}_1 = \boldsymbol{q}_1$, 则 $\boldsymbol{v}_i = \pm\boldsymbol{q}_i$ 且 $|t_{i,i-1}| = |s_{i,i-1}|, i = 2:k$. 而且若 $k < n$, 则 $s_{k+1,k} = 0$. 见定理 7.4.2.*

**证明**　定义正交矩阵 $\boldsymbol{W}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{V}$ 并注意到 $\boldsymbol{W}(:,1)=I_n(:,1)=\boldsymbol{e}_1$ 且 $\boldsymbol{W}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{W}=S$. 由定理 8.3.1, $\boldsymbol{W}^{\mathrm{T}}K(T,e_1,k)$ 是列满秩的上三角形矩阵. 但 $K(\boldsymbol{T},\boldsymbol{e}_1,k)$ 是上三角形矩阵, 这样, 由窄 QR 分解“本质上”的唯一性知

$$W(:,1:k)=I_n(:,1:k)\mathrm{diag}(\pm1,\cdots,\pm1).$$

这是说对 $i=1:k, Q(:,i)=\pm V(:,j)$. 由于 $t_{i+1,i}=Q(:,i+1)^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}(:,i)$ 和 $s_{i+1,i}=V(:,i+1)^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}(:,i), i=1:n-1$, 关于次对角元素之论述由此而知. □

### 8.3.3　QR 迭代和三对角矩阵

我们快速地叙述关于 QR 迭代和三对角矩阵的 4 个事实. 完整的证明是很简单的.

**1. 保持形式.** 若 $\boldsymbol{T}=\boldsymbol{Q}\boldsymbol{R}$ 是对称三对角矩阵 $\boldsymbol{T}\in\mathbb{R}^{n\times n}$ 的 QR 分解, 则 $\boldsymbol{Q}$ 有下带宽为 1, $\boldsymbol{R}$ 有上带宽为 2, 而且

$$\boldsymbol{T}_+=\boldsymbol{R}\boldsymbol{Q}=\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{Q}\boldsymbol{R})\boldsymbol{Q}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{Q}$$

也是对称三对角矩阵.

**2. 位移.** 若 $s\in\mathbb{R}$ 且 $\boldsymbol{T}-s\boldsymbol{I}=\boldsymbol{Q}\boldsymbol{R}$ 是 QR 分解, 则

$$\boldsymbol{T}_+=\boldsymbol{Q}\boldsymbol{R}+s\boldsymbol{I}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{Q}$$

也是三对角矩阵. 这称为位移 QR 步.

**3. 完全位移.** 若 $\boldsymbol{T}$ 是不可约的, 则无论 $s$ 值为多少, $\boldsymbol{T}-s\boldsymbol{I}$ 的前 $n-1$ 列线性无关. 这样, 如果 $s\in\lambda(\boldsymbol{T})$ 且

$$\boldsymbol{Q}\boldsymbol{R}=\boldsymbol{T}-s\boldsymbol{I}$$

是 QR 分解, 则 $r_{nn}=0$ 且 $\boldsymbol{T}_+=\boldsymbol{R}\boldsymbol{Q}+s\boldsymbol{I}$ 的最后一列等于 $sI_n(:,n)=se_n$.

**4. 运算量.** 若 $T\in\mathbb{R}^{n\times n}$ 是三对角矩阵, 则它的 QR 分解可由下列 $n-1$ 个 Givens 旋转变换计算得到:

**for** $k=1:n-1$

　　$[c,s]=\mathbf{givens}(t_{kk},t_{k+1,k})$

　　$m=\min\{k+2,n\}$

$$T(k:k+1,k:m)=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} T(k:k+1,k:m)$$

**end**

这需 $O(n)$ 个 flop. 若旋转矩阵需累积求出, 则需 $O(n^2)$ 个 flop.

### 8.3.4　显式单位移 QR 迭代

如果 $s$ 是一个很好的近似特征值, 则我们期望用位移 $s$ 进行一次 QR 迭代后 $(n,n-1)$ 元会很小. 这就是下列迭代中的基本思想:

$T=U_0^{\mathrm{T}}AU_0$　(三对角)

**for**　$k=0,1,\cdots$

$$\text{决定实位移}\mu, \tag{8.3.2}$$
$$T - \mu I = UR \quad (\text{QR分解})$$
$$T = RU + \mu I$$

**end**

如果

$$T = \begin{bmatrix} a_1 & b_1 & & \cdots & 0 \\ b_1 & a_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & & b_{n-1} & a_n \end{bmatrix},$$

则位移的一个合理选择是 $\mu = a_n$. 然而一个更有效的选择是用

$$T(n-1:n, n-1:n) = \begin{bmatrix} a_{n-1} & b_{n-1} \\ b_{n-1} & a_n \end{bmatrix}$$

靠近 $a_n$ 的特征值作位移. 这称为 Wilkinson位移, 它由

$$\mu = a_n + d - \text{sign}(d)\sqrt{d^2 + b_{n-1}^2} \tag{8.3.3}$$

给出, 其中 $d = (a_{n+1} - a_n)/2$. Wilkinson(1986b) 已证明 (8.3.2) 对上述两种位移策略都是立方收敛的, 但给出了为什么偏好 (8.3.3) 的直观理由.

### 8.3.5 隐式位移

不必显式形成矩阵 $T - \mu I$ 就能实现从 $T$ 到 $T_+ = RU + \mu I = U^{\mathrm{T}}TU$ 的变换, 这在位移比某个 $a_j$ 大很多时有优势. 令 $c = \cos(\theta)$ 和 $s = \sin(\theta)$ 使得

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_1 - \mu \\ b_1 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}.$$

如果我们令 $G_1 = G(1,2,\theta)$ 则 $G_1 e_1 = U e_1$ 且

$$T \leftarrow G_1^{\mathrm{T}} T G_1 = \begin{bmatrix} \times & \times & + & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ + & \times & \times & \times & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

这样, 只要我们能计算旋转矩阵 $G_2, \cdots, G_{n-1}$, 使得若 $Z = G_1 G_2 \cdots G_{n-1}$, 则 $Z e_1 = G_1 e_1 = U e_1$ 和 $Z^{\mathrm{T}} T Z$ 是三对角矩阵, 那么就可以应用隐式 Q 定理.

注意, 只要我们使每个 $G_i$ 是 $G_i = G(i, i+1, \theta_i), i = 2:n-1$, 则 $Z$ 和 $U$ 的第一列是相等的, 而这种形状的矩阵 $G_i$ 可用来将多余的非零元素 “+” 逐出矩阵

$G_1^{\mathrm{T}}TG_1$:

$$\xrightarrow{G_2}\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & + & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 \\ 0 & + & \times & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix} \xrightarrow{G_3}\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \times & + & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & + & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

$$\xrightarrow{G_4}\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 \\ 0 & 0 & \times & \times & \times & + \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & + & \times & \times \end{bmatrix} \xrightarrow{G_5}\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

这样, 由隐式 Q 定理可得出, 由这种逐零技巧所产生的三对角矩阵 $Z^{\mathrm{T}}TZ$ 和用显式方法获得的三对角矩阵 $T$ 本质上是一样的. (我们可以假定所有涉及的三对角矩阵都是不可约的, 否则问题可以分解).

注意, 在逐零过程的每一步, 只有一个非零元在三对角线之外. 这个非零元在修正 $T \leftarrow G_k^{\mathrm{T}}TG_k$ 下向矩阵右下方移动, 如下所示:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_k & b_k & z_k & 0 \\ b_k & a_p & b_p & 0 \\ z_k & b_p & a_q & b_q \\ 0 & 0 & b_q & a_r \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} a_k & b_k & 0 & 0 \\ b_k & a_p & b_p & z_p \\ 0 & b_p & a_q & b_q \\ 0 & z_p & b_q & a_r \end{bmatrix}$$

这里 $(p,q,r) = (k+1, k+2, k+3)$. 一旦 $c$ 和 $s$ 由等式 $b_k s + z_k c = 0$ 确定之后, 此修正大约需 26 个 flop. 总之, 我们有下列算法.

**算法 8.3.2 (带 Wilkinson 位移的隐式对称 QR 法)** 给定一不可约的对称三对角矩阵 $T \in \mathbb{R}^{n\times n}$, 本算法用 $Z^{\mathrm{T}}TZ$ 覆盖 $T$, 其中 $Z = G_1 \cdots G_{n-1}$ 是 Givens 旋转矩阵之乘积, 具有性质 $Z^{\mathrm{T}}(T-\mu I)$ 为上三角形矩阵以及 $\mu$ 是 $T$ 的右下角 $2\times 2$ 主子矩阵的特征值并且最靠近 $t_{nn}$.

$$d = (t_{n-1,n-1} - t_{nn})/2$$

$$\mu = t_{nn} - t_{n,n-1}^2/(d + \mathrm{sign}(d)\sqrt{d^2 + t_{n,n-1}^2})$$

$$x = t_{11} - \mu$$

$z = t_{21}$

**for** $k = 1 : n-1$

  $[c, s] = \textbf{givens}(x, z)$

  $T = G_k^{\mathrm{T}} T G_k$, 其中 $G_k = G(k, k+1, \theta)$

  **if** $k < n-1$

    $x = t_{k+1,k}$

    $z = t_{k+2,k}$

  **end**

**end**

这个算法需大约 $30n$ 个 flop 以及求 $n$ 个平方根. 若一给定正交矩阵 $\boldsymbol{Q}$ 被 $\boldsymbol{QG}_1 \cdots \boldsymbol{G}_{n-1}$ 覆盖, 则需额外的 $6n^2$ 个 flop. 当然, 在实际编程中, 三对角矩阵 $\boldsymbol{T}$ 应储存在两个 $n$ 维向量中而不是 $n \times n$ 维数组中.

**例 8.3.2** 如果把算法 8.3.2 应用到

$$\boldsymbol{T} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 3 & 0.01 \\ 0 & 0 & 0.01 & 4 \end{bmatrix},$$

则新的三对角矩阵 $\boldsymbol{T}$ 为

$$\boldsymbol{T} = \begin{bmatrix} 0.500\,0 & 0.591\,6 & 0 & 0 \\ 0.591\,6 & 1.785 & 0.180\,8 & 0 \\ 0 & 0.180\,8 & 3.714\,0 & 0.000\,004\,4 \\ 0 & 0 & 0.000\,004\,4 & 4.002\,497 \end{bmatrix}.$$

算法 8.3.2 是对称 QR 算法 (计算稠密对称矩阵的 Schur 分解之标准方法) 的基础.

**算法 8.3.3 (对称 QR 算法)** 给定对称矩阵 $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ 和一个比舍入误差单位大的容许误差 tol. 本算法计算一个近似对称 Schur 分解 $\boldsymbol{Q}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{Q} = \boldsymbol{D}$. $\boldsymbol{A}$ 用三对角分解所覆盖.

用算法 8.3.1, 计算三对角化

  $T = (P_1 \cdots P_{n-2})^{\mathrm{T}} A (P_1, \cdots, P_{n-2})$.

  令 $D = T$, 若需求出 $Q$, 则 $Q = P_1 \cdots P_{n-2}$. 见 5.1.6 节.

**until** $q = n$

  对 $i = 1 : n-1$, 令 $d_{i+1,i}$ 和 $d_{i,i+1}$ 为零. 若

    $|d_{i+1,i}| = |d_{i,i+1}| \leqslant \mathrm{tol}(|d_{ii}| + |d_{i+1,i+1}|)$

　　找到最大 $q$ 和最小 $p$, 使得若

$$D = \begin{bmatrix} D_{11} & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & D_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$
$$\quad\quad\ \ p \quad\quad n-p-q \quad\quad q$$

　　则 $D_{33}$ 为对角矩阵且 $D_{22}$ 是不可约的.

　　**if** $q < n$

　　　对 $D_{22}$ 应用算法 8.3.2:

$$D = \mathrm{diag}(I_p, \bar{Z}, I_q)^{\mathrm{T}} D \ \mathrm{diag}(I_p, \bar{Z}, I_q)$$

　　　若需求出 $\boldsymbol{Q}$, 则 $\boldsymbol{Q} = \boldsymbol{Q}\ \mathrm{diag}(I_p, \bar{Z}, I_q)$.

　　**end**

**end**

如果需求出 $\boldsymbol{Q}$, 这一算法约需 $9n^3$ 个 flop, 否则约需 $\frac{4}{3}n^3$ 个 flop.

**例 8.3.3**　若把算法 8.3.3 应用于三对角矩阵

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 3 & 4 & 0 \\ 0 & 4 & 5 & 6 \\ 0 & 0 & 6 & 7 \end{bmatrix}.$$

在执行算法 8.3.3 时, 次对角元素变化如下:

| 迭　代 | $a_{21}$ | $a_{32}$ | $a_{43}$ |
|---|---|---|---|
| 1 | 1.6817 | 3.2344 | 0.8649 |
| 2 | 1.6142 | 2.5755 | 0.0006 |
| 3 | 1.6245 | 1.6965 | $10^{-13}$ |
| 4 | 1.6245 | 1.6965 | 收敛 |
| 5 | 1.5117 | 0.0150 | |
| 6 | 1.1195 | $10^{-9}$ | |
| 7 | 0.7071 | 收敛 | |
| 8 | 收敛 | | |

最后, 我们得到 $\lambda(\boldsymbol{A}) = \{-2.4848, 0.7046, 4.9366, 12.831\}$.

由算法 8.3.3 得到的计算特征值 $\hat{\lambda}_i$ 是 $\boldsymbol{A}$ 附近的一个矩阵的精确特征值, 即 $\boldsymbol{Q}_0^{\mathrm{T}}(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{Q}_0 = \mathrm{diag}(\hat{\lambda}_i)$, 其中 $\boldsymbol{Q}_0^{\mathrm{T}}\boldsymbol{Q}_0 = \boldsymbol{I}$ 且 $\|\boldsymbol{E}\|_2 \approx \mathbf{u}\|\boldsymbol{A}\|_2$. 由推论 8.1.6 我们知道, 在 $|\hat{\lambda}_i - \lambda_i| \approx \mathbf{u}\|\boldsymbol{A}\|_2$ 意义下每个 $\hat{\lambda}_i$ 的绝对误差很小. 如果 $\hat{\boldsymbol{Q}} = [\hat{q}_1, \cdots, \hat{q}_n]$ 是所计算的正交特征向量组成的矩阵, 则 $\hat{q}_i$ 的精确度依赖于 $\lambda_i$ 和谱中其余元素的分离度. 见定理 8.1.12.

如果需要计算所有的特征值和部分特征向量, 则算法 8.3.3 因不累积 $\boldsymbol{Q}$ 而变的更廉价. 所需的特征向量改由 $\boldsymbol{T}$ 的逆迭代计算. 见 8.2.2 节. 通常只一步就能有效地得到一个好的特征向量, 即使初始向量是随机给的.

如果仅需少量几个特征值和特征向量, 则在 8.5 节中有一些合适的专门方法.

有趣的是, Rayleigh 商迭代和对称 QR 算法之间有一种联系. 设我们应用后者于三对角矩阵 $\boldsymbol{T} \in \mathbb{R}^{n\times n}$, 取位移 $\sigma = \boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{T}\boldsymbol{e}_n = t_{nn}$, 其中 $\boldsymbol{e}_n = I_n(:,n)$. 若 $\boldsymbol{T} - \sigma\boldsymbol{I} = \boldsymbol{QR}$, 则 $\boldsymbol{T} = \boldsymbol{RQ} + \sigma\boldsymbol{I}$. 从等式 $(\boldsymbol{T} - \sigma\boldsymbol{I})\boldsymbol{Q} = \boldsymbol{R}^{\mathrm{T}}$ 得到

$$(\boldsymbol{T} - \sigma\boldsymbol{I})\boldsymbol{q}_n = r_{nn}\boldsymbol{e}_n,$$

其中 $\boldsymbol{q}$ 是正交矩阵 $\boldsymbol{Q}$ 的最后一列. 于是, 如果我们应用 (8.2.6), 取 $\boldsymbol{x}_0 = \boldsymbol{e}_n$, 则 $\boldsymbol{x}_1 = \boldsymbol{q}_n$.

### 8.3.6 用 Ritz 加速的正交迭代

回忆 8.2.4 节可知, 正交迭代包括一个矩阵乘积和一个 QR 分解:

$$\boldsymbol{Z}_k = \boldsymbol{A}\widetilde{\boldsymbol{Q}}_{k-1}$$

$$\widetilde{\boldsymbol{Q}}_k\boldsymbol{R}_k = \boldsymbol{Z}_k \quad (\text{QR 分解})$$

定理 8.1.14 说明, 通过令 $\boldsymbol{S} = \boldsymbol{S}_k \equiv \widetilde{\boldsymbol{Q}}_k^{\mathrm{T}}\boldsymbol{A}\widetilde{\boldsymbol{Q}}_k$ 能使 $\|\boldsymbol{A}\widetilde{\boldsymbol{Q}}_k - \widetilde{\boldsymbol{Q}}_k\boldsymbol{S}\|_F$ 极小化. 如果 $\boldsymbol{U}_k^{\mathrm{T}}\boldsymbol{S}_k\boldsymbol{U}_k = \boldsymbol{D}_k$ 是 $\boldsymbol{S}_k \in \mathbb{R}^{r\times r}$ 的 Schur 分解且 $\boldsymbol{Q}_k = \widetilde{\boldsymbol{Q}}_k\boldsymbol{U}_k$, 则

$$\|\boldsymbol{A}\boldsymbol{Q}_k - \boldsymbol{Q}_k\boldsymbol{D}_k\|_F = \|\boldsymbol{A}\widetilde{\boldsymbol{Q}}_k - \widetilde{\boldsymbol{Q}}_k\boldsymbol{S}_k\|_F,$$

这表明, 从余量极小的观点来看, $k$ 步后 $\boldsymbol{Q}_k$ 的列向量是最优基. 这就定义了 Ritz加速思想:

$\boldsymbol{Q}_0 \in \mathbb{R}^{n\times P}$ 满足 $Q_0^{\mathrm{T}}Q_0 = I_p$
**for** $k = 1, 2, \cdots$
  $Z_k = AQ_{k-1}$
  $\widetilde{Q}_kR_k = Z_k$ (QR 分解)
  $S_k = \widetilde{Q}_k^{\mathrm{T}}A\widetilde{Q}_k$
  $U_k^{\mathrm{T}}S_kU_k = D_k$ (Schur 分解)
  $Q_k = \widetilde{Q}_kU_k$
**end**

能够证明, 如果

$$\boldsymbol{D}_k = \mathrm{diag}(\theta_1^{(k)}, \cdots, \theta_r^{(k)}), \quad |\theta_1^{(k)}| \geqslant \cdots \geqslant |\theta_r^{(k)}|,$$

则

$$|\theta_i^{(k)} - \lambda_i(\boldsymbol{A})| = O\left(\left|\frac{\lambda_{r+1}}{\lambda_i}\right|^k\right), \quad i = 1:r.$$

回顾定理 8.2.2 知, $\widetilde{\boldsymbol{Q}}_k^{\mathrm{T}}\boldsymbol{A}\widetilde{\boldsymbol{Q}}_k$ 的特征值以速度 $|\lambda_{r+1}/\lambda_r|^k$ 收敛. 这样, Ritz 值以更快的速度收敛, 详见 Stewart(1969).

**例 8.3.4** *若我们应用* (8.3.6) *于*

$$\boldsymbol{A} = \begin{bmatrix} 100 & 1 & 1 & 1 \\ 1 & 99 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad \boldsymbol{Q}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

则

| $k$ | $\text{dist}\{D_2(A), Q_k\}$ | $k$ | $\text{dist}\{D_2(A), Q_k\}$ |
|---|---|---|---|
| 0 | $0.2 \times 10^{-1}$ | 3 | $0.3 \times 10^{-6}$ |
| 1 | $0.5 \times 10^{-3}$ | 4 | $0.8 \times 10^{-8}$ |
| 2 | $0.1 \times 10^{-4}$ | | |

显然, 收敛速度为 $(2/99)^k$.

## 习　题

**8.3.1** 设 $\lambda$ 是对称三对角矩阵 $T$ 的特征值. 证明, 若 $\lambda$ 的代数重数为 $k$, 则 $T$ 的次对角元素至少有 $k-1$ 个为零.

**8.3.2** 设 $A$ 为对称矩阵, 且有带宽 $p$. 证明, 若我们执行带位移 QR 步 $A - \mu I = QR, A = RQ + \mu I$, 则 $A$ 有带宽 $p$.

**8.3.3** 设 $B \in \mathbb{R}^{n\times n}$ 是具有对角元 $d(1:n)$ 和超对角元 $f(1:n-1)$ 的上双对角矩阵. 陈述并证明定理 8.3.1 的奇异值形式.

**8.3.4** 令 $A = \begin{bmatrix} w & x \\ x & z \end{bmatrix}$ 为实矩阵, 假设我们执行如下的带位移 QR 步 $A - zI = UR, \bar{A} = RU + zI$. 证明: 若 $\bar{A} \begin{bmatrix} \bar{w} & \bar{x} \\ \bar{x} & \bar{z} \end{bmatrix}$, 则

$$\begin{aligned}\bar{w} &= w + x^2(x-z)/[(w-z)^2 + x^2], \\ \bar{z} &= z - x^2(w-z)/[(w-z)^2 + x^2], \\ \bar{x} &= -x^3/[(w-z)^2 + x^2].\end{aligned}$$

**8.3.5** 设 $A \in \mathbb{C}^{n\times n}$ 为 Hermit 矩阵, 说明怎样构造酉矩阵 $Q$ 使得 $Q^{\rm H}AQ = T$ 为实对称三对角矩阵.

**8.3.6** 证明若 $A = B + {\rm i}C$ 是 Hermit 矩阵, 则 $M = \begin{bmatrix} B & -C \\ C & B \end{bmatrix}$ 为对称矩阵. 试找出 $A$ 和 $M$ 的特征值与特征向量之间的关系.

**8.3.7** 对于 $A$ 储存在两个 $n$ 维向量中的情形, 重新写出算法 8.2.2. 并说明算法的计算量.

**8.3.8** 设 $A = S + \sigma uu^{\rm T}$, 其中 $S \in \mathbb{R}^{n\times n}$ 是反对称矩阵 $(A^{\rm T} = -A), u \in \mathbb{R}^n$ 为单位 2 范数向量, 且 $\sigma \in \mathbb{R}$. 说明怎样计算正交矩阵 $Q$ 使得 $Q^{\rm T}AQ$ 为三对角矩阵且 $Q^{\rm T}u = I_n(:,1) = e_1$.

## 本节注释与参考文献

对称矩阵三对角化的讨论可见:

R. S. Martin and J. H. Wilkinson (1968). "Householder's Tridiagonalization of a Symmetric Matrix," *Numer. Math. 11,* 181–195. See also Wilkinson and Reinsch (1971, pp. 212–226).

H. R. Schwartz (1968). "Tridiagonalization of a Symmetric Band Matrix," *Numer. Math.* 12, 231–241. See also Wilkinson and Reinsch (1971, pp. 273–283).

N. E. Gibbs and W. G. Poole, Jr. (1974). "Tridiagonalization by Permutations," *Comm. ACM* 17, 20–24.

前两篇参考文献含有 Algol 算法. 对于显式和隐式三对角 QR 算法的 Algol 程序见于:

H. Bowdler, R. S. Martin, C. Reinsch, and J. H. Wilkinson (1968). "The QR and QL Algorithms for Symmetric Matrices," *Numer. Math.* 11, 293–306. See also Wilkinson and Reinsch (1971, pp. 227–240).

A. Dubrulle, R. S. Martin, and J. H. Wilkinson (1968). "The Implicit QL Algorithm," *Numer. Math.* 12, 377–383. 也见 Wilkinson and Reinsch (1971, 241–248 页).

"QL" 算法等同于 QR 算法, 除了每步中矩阵 $T - \lambda I$ 分解成一个正交矩阵和一个下三角矩阵外. 含有这些方法的其他文章包括有:

G. W. Stewart (1970). "Incorporating Original Shifts into the QR Algorithm for Symmetric Tridiagonal Matrices," *Comm. ACM* 13, 365–367.

A. Dubrulle (1970). "A Short Note on the Implicit QL Algorithm for Symmetric Tridiagonal Matrices," *Numer. Math.* 15, 450.

推广到 Hermit 矩阵和反对称矩阵的讨论可见:

D. Mueller (1966). "Householder's Method for Complex Matrices and Hermitian Matrices," *Numer, Math.* 8, 72–92.

R. C. Ward and L. J. Gray (1978). "Eigensystem Computation for Skew-Symmetric and A Class of Symmetric Matrices," *ACM Trans. Math. Soft.* 4, 278–285.

算法 8.2.3 的收敛性质在 Lawson and Hanson(1974, 附录 B) 中有详述, 也可见:

J. H. Wilkinson (1968b). "Global Convergence of Tridiagonal QR Algorithm With Origin Shifts," *Lin. Alg. and Its Applic. I,* 409–420.

T. J. Dekker and J. F. Traub (1971). "The Shifted QR Algorithm for Hermitian Matrices", *Lin. Alg. and Its Applic.* 4, 137–154.

W. Hoffman and B. N. Parlett (1978). "A New Proof of Global Convergence for the Tridiagonal QL Algorithm", *SIAMJ. Num. Anal.* 15, 929–937.

S. Batterson (1994). "Convergence of the Francis Shifted QR Algorithm on Normal Matrices", *Lin. Alg. and Its Applic.* 207, 181–195.

当该方法用于一般矩阵时的分析可见:

C. P. Huang (1981). "On the Convergence of the QR Algorithm with Origin Shifts for Normal Matrices", *IMAJ. Num. Anal.* 1, 127–133.

有关三对角 QR 算法的位移的有趣论文包括:

F. L. Bauer and C. Reinsch (1968). "Rational QR Transformations with Newton Shift for Symmetric Tridiagonal Matrices", *Numer. Math.* 11, 264–272. See also Wilkinson and Reinsch (1971, pp. 257–265).

G. W. Stewart (1970). "Incorporating Origin Shifts into the QR Algorithm for Symmetric Tridiagonal Matrices", *Comm. Assoc. Comp. Mach.* 13, 365–367.

针对本节算法的一些并行计算的可行性讨论于:

S. Lo, B. Philippe, and A. Sameh (1987). "A Multiprocessor Algorithm for the Symmetric Tridiagonal Eigenvalue Problem", *SIAM J. Sci. and Stat. Comp.* 8, s155-s165.

H. Y. Chang and M. Salama (1988). "A Parallel Householder Tridiagonalization Strategy Using Scattered Square Decomposition", *Parallel Computing* 6, 297–312.

另一种计算特征值的特定子集的方法是利用智能 QR 算法. 在该方法中, 位移用 Newton 方法决定. 这能"引导"迭代趋于所需的特征值, 见:

C. Reinsch and F. L. Bauer (1968). "Rational QR Transformation with Newton's Shift for Symmetric Tridiagonal Matrices", *Numer Math.* 11, 264–272. See also Wilkinson and Reinsch (1971, pp. 257–265).

讨论带状矩阵的对称 QR 算法的文章包括:

R. S. Martin and J. H. Wilkinson (1967). "Solution of Symmetric and Unsymmetric Band Equations and the Calculation of Eigenvectors of Band Matrices", *Numer. Math.* 9, 279–301. See also Wilkinson and Reinsch (1971, pp. 70–92).

R. S. Martin, C. Reinsch, and J. H. Wilkinson (1970). "The QR Algorithm for Band Symmetric Matrices", *Numer. Math.* 16, 85–92. See also Wilkinson and Reinsch (1971, pp. 266–272).

## 8.4　Jacobi 方法

求对称特征值问题的 Jacobi 方法近来引起人们注意, 是因为它们本质上是并行的. 它们做法如下: 进行一系列正交相似变换不断校正 $\boldsymbol{A} \leftarrow \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$, 使得每个新 $\boldsymbol{A}$, 虽然是满的, 但比前一个 $\boldsymbol{A}$"更对角化". 最终, 非对角线元素都小到可以认为是零.

通过对 Jacobi 方法内在的基本思想进行观察后, 我们提出一种并行 Jacobi 过程.

### 8.4.1　Jacobi 思想

Jacobi 方法的思想是逐步地减小量

$$\mathrm{off}(\boldsymbol{A}) = \sqrt{\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n} a_{ij}^2},$$

即非对角元素的"范数". 实现此目的的工具是旋转变换:

$$\boldsymbol{J}(p,q,\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ p \\ \\ q \\ \\ \\ \end{matrix},$$

$$\qquad\qquad\qquad p \qquad\quad q$$

我们称其为 Jacobi **旋转变换**. Jacobi 旋转和 Givens 旋转没什么不同, 见 5.1.8 节. 在本节改用名字是为了纪念方法的发明者.

进行 Jacobi 特征值求解过程的基本步骤有：(1) 选择满足 $1 \leqslant p < q \leqslant n$ 的指标对 $(p,q)$; (2) 计算一个余弦–正弦对 $(c,s)$ 使得

$$\begin{bmatrix} b_{pp} & b_{pq} \\ b_{qp} & b_{qq} \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \tag{8.4.1}$$

为对角矩阵; (3) 计算 $\boldsymbol{B} = \boldsymbol{J}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{J}$, 其中 $\boldsymbol{J} = \boldsymbol{J}(p,q,\theta)$, 并覆盖 $\boldsymbol{A}$. 注意到矩阵 $\boldsymbol{B}$ 和 $\boldsymbol{A}$ 除了 $p$ 和 $q$ 这两行两列外均相等. 而且, Frobenius 范数在正交变换下不变, 我们有

$$a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2 = b_{pp}^2 + b_{qq}^2 + 2b_{pq}^2 = b_{pp}^2 + b_{qq}^2.$$

这样

$$\begin{aligned} \mathrm{off}(\boldsymbol{B})^2 &= \|\boldsymbol{B}\|_F^2 - \sum_{i=1}^{n} b_{ii}^2 \\ &= \|\boldsymbol{A}\|_F^2 - \sum_{i=1}^{n} a_{ii}^2 + (a_{pp}^2 + a_{qq}^2 - b_{pp}^2 - b_{qq}^2) \\ &= \mathrm{off}(\boldsymbol{A})^2 - 2a_{pq}^2. \end{aligned} \tag{8.4.2}$$

在此意义下, 每一个 Jacobi 步后 $\boldsymbol{A}$ 更靠近对角型.

在我们讨论怎样选取下标对 $(p,q)$ 前, 让我们看看 $(p,q)$ 子问题的实际计算过程.

### 8.4.2 2×2 的对称 Schur 分解

使 (8.4.1) 对角化意味着

$$0 = b_{pq} = a_{pq}(c^2 - s^2) + (a_{pp} - a_{qq})cs. \tag{8.4.3}$$

若 $a_{pq} = 0$, 则我们只要设 $(c,s) = (1,0)$. 否则定义

$$\tau = \frac{a_{qq} - a_{pp}}{2a_{pq}} \quad 且 \quad t = s/c.$$

由 (8.4.3) 得到 $t = \tan(\theta)$ 是二次方程 $t^2 + 2\tau t - 1 = 0$ 之解. 结果发现, 选择两根 $t = -\tau \pm \sqrt{1+\tau^2}$ 中较小的一个是重要的, 而 $c$ 和 $s$ 可由公式

$$c = 1/\sqrt{1+t^2}, \quad s = tc$$

得到. 选择 $t$ 为两根中较小的一个可确保 $|\theta| \leqslant \frac{\pi}{4}$ 且使 $\boldsymbol{B}$ 和 $\boldsymbol{A}$ 之差的范数最小, 因为

$$\|\boldsymbol{B} - \boldsymbol{A}\|_F^2 = 4(1-c) \sum_{\substack{i=1 \\ i \neq p,q}}^{n} (a_{ip}^2 + a_{iq}^2) + 2a_{pq}^2/c^2.$$

我们总结 $2 \times 2$ 矩阵计算如下.

**算法 8.4.1** *给定一个 $n \times n$ 对称矩阵 $\boldsymbol{A}$ 以及整数 $p$ 和 $q$, 满足 $1 \leqslant p < q \leqslant n$, 这一算法计算的余弦–正弦对 $(c,s)$ 使得若 $\boldsymbol{B} = \boldsymbol{J}(p,q,\theta)^{\mathrm{T}}\boldsymbol{A}\boldsymbol{J}(p,q,\theta)$, 则 $b_{pq} = b_{qp} = 0$.*

```
function:    [c,s]=sym. schur2(A,p,q)
    if  A(p,q) ≠ 0
        τ = (A(q,q) − A(p,p))/(2A(p,q))
        if  τ ≥ 0
                t = 1/(τ + √(1 + τ²));
        else
                t = −1/(−τ + √(1 + τ²));
        end
        c = 1/√(1 + t²)
        s = tc
    else
        c = 1
        s = 0
    end
```

### 8.4.3 经典 Jacobi 算法

如我们上面所说, 解决 $(p,q)$ 子问题, 只改变 $p$ 和 $q$ 这两行两列, 一旦**sym. schur2**决定了 $2\times2$ 的旋转矩阵, 那么若充分利用对称性, 则用 $\boldsymbol{J}(p,q,\theta)^{\mathrm{T}}\boldsymbol{A}\boldsymbol{J}(p,q,\theta)$ 修正 $\boldsymbol{A}$ 只需 $6n$ 个 flop.

我们怎样选择指标 $p$ 和 $q$ 呢? 从使 (8.4.2) 中的减量 $\mathrm{off}(\boldsymbol{A})$ 极小的观点看, 有理由选择 $(p,q)$ 使 $a_{pq}^2$ 极大, 这就是经典Jacobi 算法的基础.

**算法 8.4.2 (经典 Jacobi)** 给定对称矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 和容许误差 tol>0, 本算法用 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{V}$ 为正交矩阵且 $\mathrm{off}(\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V})\leqslant \mathrm{tol}\|\boldsymbol{A}\|_F$.

```
V = I_n; eps = tol||A||_F
while off (A) > eps
    选择 (p,q) 使 |a_pq| = max_{i≠j} |a_ij|
    (c,s) = sym.schur2(A,p,q)
    A = J(p,q,θ)^T A J(p,q,θ)
    V = V J(p,q,θ)
end
```

由于 $|a_{pq}|$ 是最大的非对角线元素. $\mathrm{off}(\boldsymbol{A})^2\leqslant N(a_{pq}^2+a_{qp}^2)$, 其中 $N=n(n-1)/2$. 从 (8.4.2) 可得出

$$\mathrm{off}(\boldsymbol{B})^2\leqslant\left(1-\frac{1}{N}\right)\mathrm{off}(\boldsymbol{A})^2.$$

由推导, 若 $\boldsymbol{A}^{(k)}$ 表示经 $k$ 步 Jacobi 更新后的 $\boldsymbol{A}$, 则

$$\mathrm{off}(\boldsymbol{A}^{(k)})^2\leqslant\left(1-\frac{1}{N}\right)^k\mathrm{off}(\boldsymbol{A}^{(0)})^2.$$

这证明了经典 Jacobi 算法以线性速度收敛.

然而, 该方法的渐近收敛速度比线性的要好得多. Schonhage(1964) 和 van Kempen(1966) 证明了: 只要 $k$ 足够大, 就有常数 $c$ 使得

$$\mathrm{off}(\boldsymbol{A}^{(K+N)}) \leqslant c \cdot \mathrm{off}(\boldsymbol{A}^{k})^2,$$

即二次收敛. Henrici(1958) 的较早论文就针对 $\boldsymbol{A}$ 有不同特征值的特殊情形建立了相同的结果. 在 Jacobi 迭代的收敛理论中, 条件 $|\theta| \leqslant \pi/4$ 是很关键的. 此不等式加上其他条件就排除了几乎收敛的对角元之“相互交换”的可能性. 这一点可以从由等式 (8.4.1) 所得到的公式 $b_{pp} = a_{pp} - ta_{pq}$ 和 $b_{qq} = a_{qq} + ta_{pq}$ 以及定义 $t = \sin(\theta)/\cos(\theta)$ 中看出.

习惯上称 $N$ 次 Jacobi 迭代为一次*扫描*. 这样, 在足够多步迭代后, 只要在每步扫描后考查 $\mathrm{off}(A)$, 就能观察到算法的二次收敛性.

**例 8.4.1** 应用经典 Jacobi 迭代于

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 3 & 6 & 10 \\ 1 & 4 & 10 & 20 \end{bmatrix}$$

我们有

| 扫描 | $O(\mathrm{off}(\boldsymbol{A}))$ | 扫描 | $O(\mathrm{off}(\boldsymbol{A}))$ |
|---|---|---|---|
| 0 | $10^{2}$ | 3 | $10^{-11}$ |
| 1 | $10^{1}$ | 4 | $10^{-17}$ |
| 2 | $10^{-2}$ | | |

没有严格的理论使人们预测出为获得 $\mathrm{off}(\boldsymbol{A})$ 中特定的减量所需扫描的次数. 然而, Brent and Luk(1985) 已凭经验指出扫描次数与 $\log(n)$ 成比例, 现实情形似乎也如此.

### 8.4.4 行循环算法

经典 Jacobi 方法的麻烦是每次校正要需 $O(n)$ 个 flop 算但选取最优 $(p, q)$ 却需 $O(n^2)$ 个 flop. 解决此不平衡的一条途径是将变换的顺序固定下来. 一种合理的选择是逐行对每个非对角元素进行变换. 例, 若 $n = 4$, 我们作如下循环:

$$(p, q) = (1,2), (1,3), (1,4), (2,3), (2,4), (3,4), (1,2), \cdots$$

这种排序格式称为*按行循环*, 它产生下列算法.

**算法 8.4.3 (循环 Jacobi)** 给定对称矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和容许误差 tol$> 0$, 这一算法用 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}$ 覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{V}$ 是正交矩阵且 $\mathrm{off}(\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}) \leqslant \mathrm{tol}\|\boldsymbol{A}\|_F$.

$V = I_n$

$\mathrm{eps} = \mathrm{tol}\|A\|_F$

**while** $\mathrm{off}(A) > \mathrm{eps}$

```
for p = 1 : n − 1
    for q = p + 1 : n
        (c, s) = sym.schur2(A, p, q)
        A = J(p, q, θ)^T A J(p, q, θ)
        V = V J(p, q, θ)
    end
end
end
```

循环 Jacobi 也二次收敛. (见 Wilkinson(1962) 和 van Kempen(1966).) 然而, 由于它不需进行非对角线搜索, 因此比 Jacobi 原始算法要快得多.

**例 8.4.2** *若把循环 Jacobi 方法应用到例 8.4.1 中的矩阵, 我们有*

| 扫 描 | $O(\text{off}(\boldsymbol{A}))$ | 扫 描 | $O(\text{off}(\boldsymbol{A}))$ |
|---|---|---|---|
| 0 | $10^{2}$ | 3 | $10^{-6}$ |
| 1 | $10^{1}$ | 4 | $10^{-16}$ |
| 2 | $10^{-1}$ | | |

### 8.4.5 误差分析

用 Wilkinson 误差分析, 可以证明, 若在算法 8.4.3 中需 $r$ 次扫描, 则对 $\boldsymbol{A}$ 的特征值 $\lambda_i$ 的某个排列, 计算的 $d_i$ 满足

$$\sum_{i=1}^{n}(d_i-\lambda_i)^2 \leqslant (\delta+k_r)\|\boldsymbol{A}\|_F\mathbf{u}.$$

参数 $k_r$ 温和地依赖于 $r$.

尽管循环 Jacobi 方法二次收敛, 一般来说与对称 QR 算法无法相比. 例如, 若我们只算 flop 数, 则 2 次 Jacobi 扫描粗略等于带有变换累积的 QR 算法进行约化对角型的全部计算量. 但是, 在 $n$ 较小的时候差别就不是很明显. 而且, 若一个近似特征向量矩阵 $\boldsymbol{V}$ 是已知的, 则 $\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}$ 近似于对角矩阵, 此情形适合应用 Jacobi 方法而不是 QR 方法.

Jacobi 方法的另一个有趣之处在于当 $\boldsymbol{A}$ 为正定矩阵时, 它计算的特征值相对误差较小. 为了解这一点, 注意到以上引述的 Wilkinson 分析和 8.1 节的扰动理论一起确保计算的特征值 $\hat{\lambda}_1 \geqslant \cdots \geqslant \hat{\lambda}_n$ 满足

$$\frac{|\hat{\lambda}_i-\lambda_i(\boldsymbol{A})|}{\lambda_i(\boldsymbol{A})} \approx \mathbf{u}\frac{\|\boldsymbol{A}\|_2}{\lambda_i(\boldsymbol{A})} \leqslant \boldsymbol{u}k_2(\boldsymbol{A}).$$

然而, 由 Demmel and Veselič(1992) 提出的修正的分量形式的误差分析表明, 在正定情形有

$$\frac{|\hat{\lambda}_i-\lambda_i(\boldsymbol{A})|}{\lambda_i(\boldsymbol{A})} \approx \mathbf{u}\kappa_2(\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}^{-1}), \tag{8.4.4}$$

其中 $D = \text{diag}(\sqrt{a_{11}}, \cdots, \sqrt{a_{nn}})$. 且一般来说 $\mathbf{u}\kappa_2(D^{-1}AD)$ 是一个比 $\mathbf{u}\kappa_2(A)$ 更小的近似界. 建立这一结论的关键是一些新的扰动理论和如下事实：若 $A+$ 是由当前矩阵 $A_c$ 所获得的计算 Jacobi 校正矩阵, 则 $A_+$ 的特征值在 (8.4.4) 意义下相对来说更接近 $A_c$ 的特征值. 为使整个事情合乎实际, 停止准则不基于 $\text{off}(A)$ 与 $\mathbf{u}\|A\|_F$ 之比值而基于每个 $|a_{ij}|$ 和 $\mathbf{u}\sqrt{a_{ii}a_{jj}}$ 之比值. 这一工作是具有代表性的关于高精度算法的新的研究, 它基于精细的分量形式的误差分析. 见 Mathias(1995).

### 8.4.6 并行 Jacobi 算法

求解对称特征问题的 QR 算法和 Jacobi 算法之间最有趣的区别是后者的并行本性. 为表明这点, 设 $n = 4$ 且把 6 个子问题归为如下三个*旋转集*:

$$\text{rot.set}(1) = \{(1,2),(3,4)\},$$

$$\text{rot.set}(2) = \{(1,3),(2,4)\},$$

$$\text{rot.set}(3) = \{(1,4),(2,3)\}.$$

注意到三个旋转集中每个集的所有旋转均是“非冲突的”. 即子问题 (1, 2) 和 (3, 4) 能并行处理. 与子问题 (1, 3) 和 (2, 4) 类似, 子问题 (1, 4) 和 (2, 3) 也能并行处理. 总之, 我们说

$$(i_1, j_1), (i_2, j_2), \cdots, (i_N, j_N), \quad N = (n-1)n/2$$

是集 $\{(i,j) | 1 \leqslant i < j \leqslant n\}$ 的*并行排序*, 只要对 $s = 1 : n-1$, 旋转集 $\text{rot. set}(S) = \{(i_r, j_r) : r = 1 + n(s-1)/2 : ns/2\}$ 由非冲突的旋转组成. 这需要 $n$ 为偶数, 本节里我们都假定这一点. ($n$ 为奇数时, 我们在 $A$ 的边上增加一行和一列零元素, 且在解含有这些添加的零元素子问题时十分小心就能处理.)

一种产生并行排序方法是想象一场有 $n$ 个选手的象棋比赛, 任一个选手必须与其他每一个选手各赛一场. 在 $n = 8$ 的情形, 需 7“轮”, 在第一轮中, 我们有下列四场比赛:

| 1 | 3 | 5 | 7 |
|---|---|---|---|
| 2 | 4 | 6 | 8 |

rot. set(1)={(1,2), (3,4),(5,6), (7,8)}

即 1 对 2, 3 对 4, 等等. 为安排第 2 到 7 轮比赛, 1 号选手不动, 其他选手依次移位即可.

| 1 | 2 | 3 | 5 |
|---|---|---|---|
| 4 | 6 | 8 | 7 |

rot. set(2)={(1,4), (2,6),(3,8), (5,7)}

| 1 | 4 | 2 | 3 |
|---|---|---|---|
| 6 | 8 | 7 | 5 |

rot. set(3)={(1,6), (4,8),(2,7), (3,5)}

| 1 | 6 | 4 | 2 |
|---|---|---|---|
| 8 | 7 | 5 | 3 |

rot. set(4)={(1,8), (6,7),(4,5), (2,3)}

| 1 | 8 | 6 | 4 |
|---|---|---|---|
| 7 | 5 | 3 | 2 |

rot. set(5)={(1,7), (5,8),(3,6), (2,4)}

| 1 | 7 | 8 | 6 |
|---|---|---|---|
| 5 | 3 | 2 | 4 |

rot. set(6)={(1,5), (3,7),(2,8), (4,6)}

| 1 | 5 | 7 | 8 |
|---|---|---|---|
| 3 | 2 | 4 | 6 |

rot. set(7)={(1,3), (2,3),(4,7), (6,8)}

我们能用一对整数向量 top$(1:n/2)$ 和 bot$(1:n/2)$ 来将这些操作进行编码. 在一给定轮二次中, top$(k)$ 与 bot$(k)$ 对阵, $k=1:n/2$. 下一轮的对阵安排可通过对 top 和 bot 作如下校正来获得:

```
function:[new.top, new.bot] = music(top, bot, n)
    m = n/2
    for k = 1 : m
        if k = 1
            new.top(1) = 1
        elseif k = 2
            new.top(k) = bot(1)
        elseif k > 2
            new.top(k) = top(k − 1)
        end
        if k = m
            new.bot(k) = top(k)
        else
            new.bot(k) = bot(k + 1)
        end
    end
```

用 music 程序, 我们获得下列的并行排序 Jacobi 算法.

**算法 8.4.4 (并行排序 Jacobi)** 给定一对称矩阵 $A\in\mathbb{R}^{n\times n}$ 和容许误差 tol> 0, 本算法计算 $V^{\mathrm{T}}AV$ 并覆盖 $A$, 其中 $V$ 是正交矩阵且 off$(V^{\mathrm{T}}AV)\leqslant$ tol$\|A\|_F$. 假定 $n$ 为偶数.

```
V = I_n
eps = tol||A||_F
top = 1 : 2 : n; bot = 2 : 2 : n
while  off(A) > eps
      for  set = 1 : n − 1
```

```
            for k = 1 : n/2
                p = min(top(k), bot(k))
                q = max(top(k), bot(k))
                (c, s) = sym.schur2(A, p, q)
                A = J(p, q, θ)^T A J(p, q, θ)
                V = V J(p, q, θ)
            end
            [top, bot] = music(top, bot, n)
        end
    end
```

注意到 $k$ 循环执行的是 $n/2$ 个独立的、非冲突的子问题.

### 8.4.7 环过程

我们现在讨论算法 8.4.4 是怎样用于一个 $p$ 个处理器的环. 为清楚见, 我们假设 $p = n/2$. 在任何瞬间, $\text{Proc}(\mu)$ 包含 $\boldsymbol{A}$ 的两列和 $\boldsymbol{V}$ 的相应列. 例如, 若 $n = 8$, 则以下是 $\boldsymbol{A}$ 的列之分布逐步变化的情况.

| | Proc(1) | Proc(2) | Proc(3) | Proc(4) |
|---|---|---|---|---|
| 第一步： | [1 2] | [3 4] | [5 6] | [7 8] |
| 第二步： | [1 4] | [2 6] | [3 8] | [5 7] |
| 第三步： | [1 6] | [4 8] | [2 7] | [3 5] |

顺序对表示所选的列之指标. 第一个指标称为*左列*, 第二个指标称为*右列*. 这样, 在第 3 步的 Proc(3) 中的左列和右列分别是 2 和 7.

注意到在步与步之间, 列与列按照**music**程序所隐含的置换不断交换且变换主要是在相邻的指标之间. 在每一步, 每个处理器解决一个子问题. 这包括 (a) 计算一个正交矩阵 $\boldsymbol{V}_{\text{small}} \in \mathbb{R}^{2\times 2}$, 它求解一个局部 $2\times 2$ Schur 问题, (b) 用 $2\times 2$ $\boldsymbol{V}_{\text{small}}$ 校正 $\boldsymbol{A}$ 和 $\boldsymbol{V}$ 的两选定列, (c) 送 $2\times 2$ $\boldsymbol{V}_{\text{small}}$ 到所有别的处理器, (d) 从别的处理器接收 $\boldsymbol{V}_{\text{small}}$ 并且校正 $\boldsymbol{A}$ 和 $\boldsymbol{V}$ 的局部块. 由于 $\boldsymbol{A}$ 按列储存, 校正 $\boldsymbol{V}_{\text{small}}$ 需要进行信息传递, 因为他们影响 $\boldsymbol{A}$ 的行. 例如, 在 $n = 8$ 的问题第二步中, Proc(2) 必须接收与子问题 (1, 4), (3, 8) 和 (5, 7) 有关的 $2\times 2$ 旋转矩阵. 这些分别来自 Proc(1), Proc(3) 和 Proc(4). 一般来说, 能够很方便地实现分享旋转矩阵, 只要通过按轮流方式在环上传递 $2\times 2$ $\boldsymbol{V}_{\text{small}}$ 矩阵. 每个处理器拷贝一份传来的 $2\times 2$ $\boldsymbol{V}_{\text{small}}$ 矩阵放在本地内存里, 然后相应校正 $\boldsymbol{A}$ 和 $\boldsymbol{V}$ 的局部选定块.

算法 8.4.4 的终止准则引出了分布式内存环境中的一个问题, 因为 $\text{off}(\cdot)$ 的值和 $\|\boldsymbol{A}\|_F$ 需要 $\boldsymbol{A}$ 的所有元素. 然而, 这些全局量在 $\boldsymbol{V}$ 矩阵轮流坐桩状态时可计算. 在开始进行 $\boldsymbol{V}$ 循环之前, 每个处理器都计算 $\|\boldsymbol{A}\|_F$ 和 $\text{off}(\cdot)$ 的相应部分. 将这些量进行轮转且每一步将其读下, 则它们能被每个处理器相加. 一个完整的循环结束之后, 每个处理器有它自己的备份 $\|\boldsymbol{A}\|_F$ 和 $\text{off}(\cdot)$.

### 8.4.8 分块 Jacobi 算法

常碰到这样的情形：在一个有 $p$ 个处理器 $(n \gg p)$ 的机器上解一个对称特征值问题. 在这种情形, Jacobi 算法的分块形式也许更合适. 上面程序的分块形式是显然的. 设 $n = rN$ 且我们对 $n \times n$ 矩阵 $\boldsymbol{A}$ 作如下划分：

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \cdots & \boldsymbol{A}_{1N} \\ \vdots & & \vdots \\ \boldsymbol{A}_{N1} & \cdots & \boldsymbol{A}_{NN} \end{bmatrix},$$

这里每个 $\boldsymbol{A}_{ij}$ 是 $r \times r$ 矩阵. 在分块 Jacobi 算法中 $(p, q)$ 子问题包含计算 $2r \times 2r$ Schur 分解

$$\begin{bmatrix} \boldsymbol{V}_{pp} & \boldsymbol{V}_{pq} \\ \boldsymbol{V}_{qp} & \boldsymbol{V}_{qq} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{A}_{pp} & \boldsymbol{A}_{pq} \\ \boldsymbol{A}_{qp} & \boldsymbol{A}_{qq} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_{pp} & \boldsymbol{V}_{pq} \\ \boldsymbol{V}_{qp} & \boldsymbol{V}_{qq} \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_{pp} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{D}_{qq} \end{bmatrix}$$

然后用由 $\boldsymbol{V}_{ij}$ 构成的分块 Jacobi 旋转矩阵作用于 $\boldsymbol{A}$. 若我们称此分块旋转矩阵为 $\boldsymbol{V}$, 则易证

$$\mathrm{off}(\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V})^2 = \mathrm{off}(\boldsymbol{A})^2 - (2\|\boldsymbol{A}_{pq}\|_F^2 + \mathrm{off}(\boldsymbol{A}_{pp})^2 + \mathrm{off}(\boldsymbol{A}_{qq})^2).$$

分块 Jacobi 方法有许多有趣的计算问题. 例如, 有许多方法去解这些子问题, 而选择看来是很重要的. 见 Bischof(1987).

## 习 题

**8.4.1** 设标量 $\gamma$ 和矩阵

$$\boldsymbol{A} = \begin{bmatrix} w & x \\ x & z \end{bmatrix}$$

一起给定, 需求一个正交矩阵

$$\boldsymbol{J} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix},$$

使得 $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{J}$ 的 (1, 1) 元素等于 $\gamma$. 证明, 此要求导致等式

$$(w - v)\tau^2 - 2x\tau + (z - \gamma) = 0,$$

其中 $\tau = c/s$. 证明只要 $\gamma$ 满足 $\lambda_2 \leqslant \gamma \leqslant \lambda_1$ 此二次方程就有实根, 其中 $\lambda_1$ 和 $\lambda_2$ 为 $\boldsymbol{A}$ 的特征值.

**8.4.2** 令 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为对称矩阵. 给出一算法计算分解

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \gamma\boldsymbol{I} + \boldsymbol{F},$$

其中 $\boldsymbol{Q}$ 是 Jacobi 旋转矩阵的累积, $\gamma = \mathrm{trace}(\boldsymbol{A})/n$, 且 $\boldsymbol{F}$ 的对角元为零. 讨论 $\boldsymbol{Q}$ 的唯一性.

**8.4.3** 对 (a) 反对称矩阵和 (b) 复 Hermit 矩阵情形设计 Jacobi 程序.

**8.4.4** 对 $n \times n$ 实对称矩阵 $\boldsymbol{A}$ 作如下划分：

$$\boldsymbol{A} = \begin{bmatrix} a & v^{\mathrm{T}} \\ v & \boldsymbol{A}_1 \end{bmatrix} \begin{matrix} 1 \\ n-1 \end{matrix}$$
$$\qquad\quad \begin{matrix} 1 & \quad n-1 \end{matrix}$$

令 $Q$ 为 Householder 矩阵, 使得若 $B = Q^{\mathrm{T}}AQ$, 则 $B(3:n,1) = 0$. 令 $J = J(1,2,\theta)$ 这样确定: 若 $C = J^{\mathrm{T}}BJ$, 则 $c_{12} = 0$ 且 $c_{11} \geqslant c_{22}$. 证明 $c_{11} \geqslant a + \|v\|_2$. 基于重复做这个 Householder-Jacobi 计算, La Budde(1964) 构造了一个求解对称特征值问题的算法.

**8.4.5** 设计函数**music**使得它需要的工作空间极小.

**8.4.6** 当应用循环 Jacobi 方法时, 跳过 $a_{pq}$ 的消去是合理的, 只要其模小于某一个小的、依赖扫描的参数, 因为在此情况下 off($A$) 的净下降量并不值得进行变换. 这就导致了所谓的门限 Jacobi 方法. 关于 Jacobi 算法的这一变形之细节可在 Wilkinson(1965, 277 页) 中找到. 证明合适的门限能保证收敛.

## 本节注释与参考文献

Jacobi 的原创性论文是在数值分析文献中能找到的最早参考文献之一:

C. G. J. Jacobi(1846). "Uber ein Leichtes Verfahren Die in der Theorie der Sacularstroungen Vorkommendern Gleichungen Numerisch Aufzulosen," *Crelle's. J.* 30,51–94.

在 QR 算法之前, Jacobi 技巧是求解稠密的对称特征值问题的标准方法, 最早的对之进行改进的一些尝试包括有:

M. Lotkin (1956). "Characteristic Values of Arbitrary Matrices," *Quart. Appl. Math.* 14, 267–275.

D. A. Pope and C. Tompkins (1957). "Maximizing Functions of Rotations: Experiments Concerning Speed of Diagonalization of Symmetric Matrices Using Jacobi's Method," *J. ACM* 4, 459–466.

C. D. La Budde (1964). "Two Classes of Algorithms for Finding the Eigenvalues and Eigenvectors of Real Symmetric Matrices," *J. ACM* 11, 53–58.

Wilkinson (1965, 265 页) 描述了 Jacobi 方法的计算特性. 也请见:

H. Rutishauser (1966). "The Jacobi Method for Real Symmetric Matrices," *Numer. Math.* 9, 1–10. See also Wilkinson and Reinsch (1971, pp. 202–211).

N. Mackey (1995). "Hamilton and Jacobi Meet Again: Quaternions and the Eigenvalue Problem," *SIAMJ. Matrix Anal. Applic.* 16, 421–435.

该方法在对角化一个几乎对角化的矩阵也是很有用的, 见:

J. H. Wilkinson (1968). "Almost Diagonal Matrices with Multiple or Close Eigenvalues," *Lin. Alg. and Its Applic. I,* 1–12

建立经典 Jacobi 迭代和循环 Jacobi 迭代的二次收敛性已经引起许多关注:

P. Henrici(1958). "On the Speed of Convergence of Cyclic and Quasicyclic Jacobi Methods for Computing the Eigenvalues of Hermitian Matrices," *SIAM J. Appl. Math.* 6, 144-162.

E. R. Hansen (1962). "On Quasicyclic Jacobi Methods," *ACM J.* 9, 118–135.

J. H. Wilkinson (1962). "Note on the Quadratic Convergence of the Cyclic Jacobi Process," *Numer. Math.* 6, 296–300.

E. R. Hansen (1963). "On Cyclic Jacobi Methods," *SIAM J. Appl. Math.* 11, 448–459.

A. Schonhage (1964). "On the Quadratic Convergence of the Jacobi Process," *Numer. Math.* 6, 410–412.

H. P. M. van Kempen (1966). "On Quadratic Convergence of the Special Cyclic Jacobi Method," *Numer. Math.* 9, 19–22.

P. Henrici and K. Zimmermann (1968). "An Estimate for the Norms of Certain Cyclic Jacobi Operators," *Lin. Alg. and Its Applic,* 1, 489–501.

K. W. Brodlie and M. J. D. Powell (1975). "On the Convergence of Cyclic Jacobi Methods," *J. Inst. Math. Applic.* 15, 279–287.

建立重要的、按元素的误差界之详尽的误差分析包括:

J. Barlow and J. Demmel (1990). "Computing Accurate Eigensystems of Scaled Diagonally Dominant Matrices," *SIAM J. Numer. Anal.* 27, 762–791.

J. W. Demmel and K. Veseliĉ (1992). "Jacobi's Method is More Accurate than QR," *SIAM J. Matrix Anal. Appl.* 13, 1204–1245.

Z. Drmač(1994). *The Generalized Singular Value Problem,* Ph. D. Thesis, Fern Universitat, Hagen, Germany.

W. F. Mascarenhas (1994). "A Note on Jacobi Being More Accurate than QR," *SIAM J. Matrix Anal. Appl.* 15, 215–218.

R. Mathias (1995). "Accurate Eigensystem Computations by Jacobi Methods," *SIAM J. Matrix Anal. Appl.* 16, 977–1003.

人们已经尝试将 Jacobi 迭代推广到其他矩阵类并且推出相应的收敛结果. 正规矩阵情形可见:

H. H. Goldstine and L. P. Horowitz (1959). "A Procedure for the Diagonalization of Normal Matrices," *J. Assoc. Comp. Mach.* 6, 176–195.

G. Loizou (1972), "On the Quadratic Convergence of the Jacobi Method for Normal Matrices," *Comp.* j. 15, 274–276.

A. Ruhe (1972). "On the Quadratic Convergence of the Jacobi Method for Normal Matrices," *BIT* 7, 305–313.

也见:

M. H. C. Paardekooper (1971). "An Eigenvalue Algorithm for Skew Symmetric Matrices," *Numer. Math.* 17, 189–202.

D. Hacon (1993). "Jacobi's Method for Skew-Symmetric Matrices," *SIAM J. Matrix Anal. Appl.* 14, 619–628.

本质上, 书中提供的分析和算法发展做少许变化就能用于正规情形. 对于非正规矩阵, 情形相当困难, 参阅:

J. Greenstadt (1955). "A Method for Finding Roots of Arbitrary Matrices," *Math. Tables and Other Aids to Comp.* 9, 47–52.

C, E, Froberg (1965). "On Triangularization of Complex Matrices by Two Dimensional Unitary Tranformations," *BIT* 5, 230–234.

J. Boothroyd and P. J. Eberlein (1968). "Solution to the Eigenproblem by a Norm-Reducing Jacobi-Type Method (Handbook)," *Numer. Math.* 11, 1–12. See also Wilkinson and Reinsch (1971, pp. 327–338).

A. Ruhe (1968). "On the Quadratic Convergence of a Generalization of the Jacobi Method to Arbitrary Matrices," *BIT* 8, 210–231.

A. Ruhe (1969). "The Norm of a Matrix After a Similarity Transformation," *BIT* 9, 53–58.

P. J. Eberlein (1970). "Solution to the Complex Eigenproblem by a Norm-Reducing Jacobi-

type Method," *Numer. Math.* 14, 232–245. See also Wilkinson and Reinsch (1971, pp. 404–417).

C. P. Huang (1975). "A Jacobi-Type Method for Triangularzing an Arbitrary Matrix," *SIAM J. Num. Anal.* 12, 566–570.

V. Hari (1982). "On the Global Convergence of the Eberlein Method for Real Matrices," *Numer. Math.* 39, 361–370.

G. W. Stewart (1985). "A Jacobi-Like Algorithm for Computing the Schur Decomposition of a Nonhermitian Matrix," *SIAM J. Sci. and Stat. Comp.* 6, 853–862.

W-W. Lin and C. W. Chen (1991). "An Acceleration Method for Computing the Generalized Eigenvalue Problem on a Parallel Computer," *Lin. Alg. and Its Applic.* 146, 49–65.

对于复对称矩阵的 Jacobi 方法也已经得到发展, 见:

J. J. Seaton (1969). "Diagonalization of Complex Symmetric Matrices Using a Modified Jacobi Methd," *Comp. J.* 12, 156–157.

P. J. Eberlein (1971). "On the Diagonalization of Complex Symmetric Matrices," *J. Inst. Math. Applic.* 7, 377–383.

P. Anderson and G. Loizou (1973). "On the Quadratic Convergence of an Algorithm Which Diagonalizes a Complex Symmetric Matrix," *J. Inst. Math. Applic.* 12, 261–271.

P. Anderson and G. Loizou (1976). "A Jacobi-Type Method for Complex Symmetric Matrices (Handbook)," *Numer. Math.* 25, 347–363.

尽管对称 QR 算法一般来说比 Jacobi 方法要快, 但在一些特定环境下, 后者更能引起人们的兴趣. 正像我们所说明的, 在一个并行计算机上同时做几个变换是可能的, 因此加快非对角元素的约化. 见:

A. Sameh (1971). "On Jacobi and Jacobi-like Algorithms for a Parallel Computer," *Math. Comp.* 25, 579–590.

J. J. Modi and J. D. Pryce (1985). "Efficient Implementation of Jacobi's Diagonalization Method on the DAP," *Numer. Math.* 46, 443-454.

D. S. Scott, M. T. Heath, and R. C. ward (1986). "Parallel Block Jacobi Eigenvalue Algorithms Using Systolic Arrays," *Lin Alg. and Its Applic.* 77, 345–356.

P. J. Eberlein (1987). "On Using the Jacobi Method on a Hypercube," in *Hypercube Multiprocessors,* ed. M. T. Heath, SIAM Publications, Philadelphia.

G. Shroff and R. Schreiber (1989). "On the Convergence of the Cyclic Jacobi Method for Parallel Block Orderings," *SIAM J. Matrix Anal. Appl.* 10, 326–346.

M H. C. Paardekooper (1991). "A Quadratically Convergent Parallel Jacobi Process for Diagonally Dominant Matrices with Nondistinct Eigenvalues," *Lin. Alg. and Its Applic.* 145, 71–88.

## 8.5 三对角方法

在这一节里, 我们阐述求对称三对角型特征问题的特殊方法. 三对角型

$$T = \begin{bmatrix} a_1 & b_1 & & \cdots & 0 \\ b_1 & a_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & & b_{n-1} & a_n \end{bmatrix} \tag{8.5.1}$$

能通过 Householder 约化 (见 8.3.1 节) 获得. 然而, 在许多情形, 很自然就出现对称三对角特征问题.

我们首先讨论二分法, 此法在只需求特征系统选定部分时很有趣. 之后给出分而治之算法, 该算法可用来以一种适于并行处理的方式来求得满对称 Schur 分解.

### 8.5.1 二分法求特征值

令 $\boldsymbol{T}_r$ 表示 (8.5.1) 式中矩阵 $\boldsymbol{T}$ 的前 $r \times r$ 阶主子矩阵. 定义多项式 $p_r(x) = \det(\boldsymbol{T}_r - x\boldsymbol{I}), r = 1:n$. 利用简单的行列式展开可以证明: 若令 $p_0(x) = 1$, 则对 $r = 2:n$, 有

$$p_r(x) = (a_r - x)p_{r-1}(x) - b_{r-1}^2 p_{r-2}(x). \tag{8.5.2}$$

因在 $O(n)$ 个 flop 内就可以计算 $p_n(x)$ 的值, 因此用半分法找它的根是可行的. 例如, 若 $p_n(y)p_n(z) < 0$ 且 $y < z$, 则迭代

```
while |y − z| > ε(|y| + |z|)
    x = (y + z)/2
    if pₙ(x)pₙ(y) < 0
        z = x
    else
        y = x
    end
end
```

保证终止, 且 $(y+z)/2$ 是 $p_n(x)$ 的一个近似零点, 即 $\boldsymbol{T}$ 的近似特征值. 这个迭代线性收敛: 它每步将误差大约减半.

### 8.5.2 Sturm 序列方法

有时需要对某个给定的 $k$, 计算 $\boldsymbol{T}$ 的第 $k$ 大的特征值. 利用对分思想和如下经典结果可有效地做到这一点.

**定理 8.5.1 (Sturm 序列性质)** 若 (8.5.1) 中的三对角矩阵 $\boldsymbol{T}$ 没有零次对角元, 则 $\boldsymbol{T}_{r-1}$ 的特征值严格分离 $\boldsymbol{T}_r$ 的特征值:

$$\lambda_r(\boldsymbol{T}_r) < \lambda_{r-1}(\boldsymbol{T}_{r-1}) < \lambda_{r-1}(\boldsymbol{T}_r) < \cdots < \lambda_2(\boldsymbol{T}_r) < \lambda_1(\boldsymbol{T}_{r-1}) < \lambda_1(\boldsymbol{T}_r).$$

此外, 若 $a(\lambda)$ 表示在序列 $\{p_0(\lambda), p_1(\lambda), \cdots, p_n(\lambda)\}$ 中符号改变个数, 则 $a(\lambda)$ 等于 $\boldsymbol{T}$ 的比 $\lambda$ 小的特征值个数. 这里多项式 $p_r(x)$ 由 (8.5.2) 所定义且我们约定若 $p_r(\lambda) = 0$, 则 $p_r(\lambda)$ 与 $p_{r-1}(\lambda)$ 反号.

**证明** 由定理 8.1.7 得出 $\boldsymbol{T}_{r-1}$ 的特征值弱分离 $\boldsymbol{T}_r$ 的特征值. 为证明必定是严格分离, 设对某些 $r$ 和 $\mu, p_r(\mu) = p_{r-1}(\mu) = 0$, 则由 (8.5.2) 及 $\boldsymbol{T}$ 的不可约性得出 $p_0(\mu) = p_1(\mu) = \cdots = p_r(\mu) = 0$, 显然矛盾. 这样, 必然有严格分离.

关于 $a(\lambda)$ 的这个断言是由 Wilkinson(1965, 300~301 页) 建立的. 我们提到, 若 $p_r(\lambda) = 0$, 则约定它的符号是与 $p_{r-1}(\lambda)$ 的符号相反. □

**例 8.5.1** 若

$$\boldsymbol{T} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix},$$

则 $\lambda(\boldsymbol{T}) \approx \{0.254, 1.82, 3.18, 4.74\}$. 序列

$$\{p_0(2), p_1(2), p_2(2), p_3(2), p_4(2)\} = \{1, -1, -1, 0, 1\}$$

证实了有两个比 $\lambda = 2$ 小的特征值.

假定我们想要计算 $\lambda_k(\boldsymbol{T})$. 由 Gershgorin 定理 (定理 8.1.3) 得出 $\lambda_k(\boldsymbol{T}) \in [y, z]$, 其中

$$y = \min_{1 \leqslant i \leqslant n} a_i - |b_i| - |b_{i-1}|, z = \max_{1 \leqslant i \leqslant n} a_i + |b_i| + |b_{i-1}|.$$

若我们定义 $b_0 = b_n = 0$. 以此为初始值, 显然由 Sturm 序列的性质确知, 迭代

$$\begin{array}{l} \textbf{while } |z-y| > \mathbf{u}(|y|+|z|) \\ \quad x = (y+z)/2 \\ \quad \textbf{if } a(x) \geqslant n-k \\ \qquad z = x \\ \quad \textbf{else} \\ \qquad y = x \\ \quad \textbf{end} \\ \textbf{end} \end{array} \tag{8.5.3}$$

产生一个长度反复减半而且始终包含 $\lambda_k(\boldsymbol{T})$ 的子区间序列.

**例 8.5.2** 若对 $k = 3$ 将 (8.5.3) 用到例 8.5.1 的矩阵, 则可得到下表所示之值

| $y$ | $z$ | $x$ | $a(x)$ |
|---|---|---|---|
| 0.0000 | 5.0000 | 2.5000 | 2 |
| 0.0000 | 2.5000 | 1.2500 | 1 |
| 1.2500 | 2.5000 | 1.3750 | 1 |
| 1.3750 | 2.5000 | 1.9375 | 2 |
| 1.3750 | 1.9375 | 1.6563 | 1 |
| 1.6563 | 1.9375 | 1.7969 | 1 |

由此可知 $\lambda_3(\boldsymbol{T}) \in [1.7969, 1.9375]$. 注意: $\lambda_3(\boldsymbol{T}) \approx 1.82$.

在执行迭代 (8.5.3) 时, 可以得到关于其他特征值大约位置的信息. 通过系统地保存这个信息能设计一个有效的方案来计算 $\lambda(\boldsymbol{T})$ 的"邻接"子集合, 例如, $\lambda_k(\boldsymbol{T})$, $\lambda_{k+1}(\boldsymbol{T}), \cdots, \lambda_{k+j}(\boldsymbol{T})$, 见 Barth, Martin, and Wilkinson(1967).

若想要一般对称矩阵 $\boldsymbol{A}$ 的一些选定特征值, 那么, 在应用上面的二分法之前就必须先计算三对角化 $\boldsymbol{T} = \boldsymbol{U}_0^{\mathrm{T}}\boldsymbol{T}\boldsymbol{U}_0$. 这个约化使用算法 8.3.1 或下一章讨论的 Lanczos 算法都可做到. 两种情形都可以经过逆迭代容易地找到相应的特征向量, 因为在 $O(n)$ 个 flop 内就可找出三对角方程组的解, 见 4.3.6 节和 8.2.2 节.

在一些应用中, 原矩阵已经是三对角型, 用二分法计算出的特征值不论其大小, 都具有较小的相对误差. 这与三对角 QR 迭代不同, 后者仅保证计算的特征值 $\widetilde{\lambda}_i$ 具有较小的绝对误差: $|\widetilde{\lambda}_i - \lambda_i(\boldsymbol{T})| \approx \mathbf{u}\|\boldsymbol{T}\|_2$.

最后, 用 $\mathrm{LDL^T}$ 分解 (见 4.2 节) 能计算对称矩阵的某些指定的特征值. 其思想是用 Sylvester 惯性定理 (定理 8.1.17). 若

$$\boldsymbol{A} - \mu\boldsymbol{I} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}, \quad \boldsymbol{A} = \boldsymbol{A}^{\mathrm{T}} \in \mathbb{R}^{n\times n}$$

是 $\boldsymbol{A} - \mu\boldsymbol{I}$ 的 $\mathrm{LDL^T}$ 分解, 且 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n)$, 则负的 $d_i$ 个数等于比 $\mu$ 小的 $\lambda_i(\boldsymbol{A})$ 个数. 详见 Parlett(1980, 46 页).

### 8.5.3 对角矩阵和秩 1 矩阵的特征问题

求对称三对角特征问题的下一个方法需要我们能够有效计算形如 $\boldsymbol{D} + \rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}}$ 的矩阵的特征值和特征向量, 其中 $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ 为对角矩阵, $\boldsymbol{z} \in \mathbb{R}^n$ 且 $\rho \in \mathbb{R}$, 这个问题就其本身来说很重要, 且主要计算依赖下面的两个结果.

**引理 8.5.2** 假定 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n) \in \mathbb{R}^{n\times n}$ 具有性质 $d_1 > \cdots > d_n$, 又设 $\rho \neq 0$ 且 $\boldsymbol{z} \in \mathbb{R}^n$ 没有零分量. 若

$$(\boldsymbol{D} + \rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}})\boldsymbol{v} = \lambda\boldsymbol{v}, \quad \boldsymbol{v} \neq \boldsymbol{0},$$

则 $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v} \neq 0$, 且 $\boldsymbol{D} - \lambda\boldsymbol{I}$ 非奇异.

**证明** 若 $\lambda \in \lambda(\boldsymbol{D})$, 则对某个 $i, \lambda = d_i$, 这样

$$0 = \boldsymbol{e}_i^{\mathrm{T}}[(\boldsymbol{D} - \lambda\boldsymbol{I})\boldsymbol{v} + \rho(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v})\boldsymbol{z}] = \rho(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v})z_i.$$

由于 $\rho$ 和 $z_i$ 非零, 我们一定有 $0 = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{v}$ 且 $\boldsymbol{D}\boldsymbol{v} = \lambda\boldsymbol{v}$. 然而, $\boldsymbol{D}$ 有相异特征值, 因此, $\boldsymbol{v} \in \mathrm{span}\{\boldsymbol{e}_i\}$. 但是 $0 = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{v} = z_i$, 矛盾. 这样, $\boldsymbol{D}$ 和 $\boldsymbol{D} + \rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}}$ 没有任何相同的特征值, 且 $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v} \neq 0$. □

**定理 8.5.3** 设 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_n) \in \mathbb{R}^{n\times n}$ 且对角元满足 $d_1 > \cdots d_n$, 假定 $\rho \neq 0$ 且 $z \in \mathbb{R}^n$ 没有零分量, 若 $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ 为正交矩阵, 使得

$$\boldsymbol{V}^{\mathrm{T}}(\boldsymbol{D} + \rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}})\boldsymbol{V} = \mathrm{diag}(\lambda_1, \cdots, \lambda_n),$$

其中 $\lambda_1 \geqslant \cdots \geqslant \lambda_n$ 且 $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n]$, 则

(a) $\lambda_i$ 是 $f(\lambda) = 1 + \rho\boldsymbol{z}^{\mathrm{T}}(\boldsymbol{D} - \lambda\boldsymbol{I})^{-1}\boldsymbol{z}$ 的 $n$ 个零解

(b) 若 $\rho > 0$, 则 $\lambda_1 > d_1 > \lambda_2 > \cdots > d_n$.

若 $\rho < 0$, 则 $d_1 > \lambda_1 > d_2 > \cdots > d_n > \lambda_n$.

(c) 特征向量 $v_i$ 是 $(\boldsymbol{D}-\lambda_i \boldsymbol{I})^{-1}\boldsymbol{z}$ 的倍数.

**证明** 若 $(\boldsymbol{D}+\rho \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}})\boldsymbol{v}=\lambda \boldsymbol{v}$, 则

$$(\boldsymbol{D}-\lambda \boldsymbol{I})\boldsymbol{v}+\rho(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v})\boldsymbol{z}=0. \tag{8.5.4}$$

我们从引理 8.5.2 知 $\boldsymbol{D}-\lambda \boldsymbol{I}$ 非奇异. 这样,

$$\boldsymbol{v}\in \operatorname{span}\{(\boldsymbol{D}-\lambda \boldsymbol{I})^{-1}\boldsymbol{z}\},$$

因此成立 (c). 而且, 若我们在等式 (8.5.4) 两边同乘以 $\boldsymbol{z}^{\mathrm{T}}(\boldsymbol{D}-\lambda \boldsymbol{I})^{-1}$ 则有

$$\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v}(1+\rho \boldsymbol{z}^{\mathrm{T}}(\boldsymbol{D}-\lambda \boldsymbol{I})^{-1}\boldsymbol{z})=0.$$

由引理 8.5.2, $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{v}\neq 0$, 这样就证明了若 $\lambda\in\lambda(\boldsymbol{D}+\rho \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}})$, 则 $f(\lambda)=0$. 我们必须证明 $f$ 的所有零点是 $\boldsymbol{D}+\rho \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}}$ 的特征值且相互交错的关系式 (b) 成立.

为达此目的, 我们仔细看看等式

$$f(\lambda)=1+\rho\left(\frac{z_1^2}{d_1-\lambda}+\cdots+\frac{z_n^2}{d_n-\lambda}\right),$$

$$f'(\lambda)=\rho\left(\frac{z_1^2}{(d_1-\lambda)^2}+\cdots+\frac{z_n^2}{(d_n-\lambda)^2}\right).$$

注意到 $f$ 在它的极点之间是单调的. 这使我们得到结论, 若 $\rho>0$, 则 $f$ 恰有 $n$ 个根, 在以下每个区间里有一个

$$(d_n,d_{n-1}),\cdots,(d_2,d_1),(d_1,\infty).$$

若 $\rho<0$, 则 $f$ 正好有 $n$ 个根, 在以下每个区间有一个

$$(-\infty,d_n),(d_n,d_{n-1}),\cdots,(d_n,d_1).$$

在两种情形下, 可知道 $f$ 的根正好是 $\boldsymbol{D}+\rho \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$ 的特征值. □

定理表明, 要计算 $\boldsymbol{V}$, 我们应 (a) 用牛顿型算法找出 $f$ 的根 $\lambda_1,\cdots\lambda_n$ 且 (b) 对 $i=1:n$, 通过正规化向量 $(\boldsymbol{D}-\lambda_i\boldsymbol{I})^{-1}\boldsymbol{z}$ 来计算 $\boldsymbol{V}$ 的列向量. 即使有重复的 $d_i$ 和零分量 $z_i$, 也能得到同样的求解方法.

**定理 8.5.4** 若 $\boldsymbol{D}=\operatorname{diag}(d_1,\cdots,d_n)$ 且 $\boldsymbol{z}\in\mathbb{R}^n$, 则存在一正交矩阵 $\boldsymbol{V}_1$ 使得若 $\boldsymbol{V}_1^{\mathrm{T}}\boldsymbol{D}\boldsymbol{V}_1=\operatorname{diag}(\mu_1,\cdots,\mu_n)$ 且 $\boldsymbol{w}=\boldsymbol{V}_1^{\mathrm{T}}\boldsymbol{z}$ 则

$$\mu_1>\mu_2>\cdots>\mu_r\geqslant\mu_{r+1}\geqslant\cdots\geqslant\mu_n,$$

对 $i=1:r, w_i\neq 0$. 且对 $i=r+1:n, w_i=0$.

**证明** 我们给出一个构造性证明, 它基于两个基本运算. (a) 假设对某个 $i<j$ 有 $d_i=d_j$. 令 $\boldsymbol{J}(i,j,\theta)$ 是在 $(i,j)$ 平面上的旋转变换, 具有性质 $\boldsymbol{J}(i,j,\theta)^{\mathrm{T}}\boldsymbol{z}$ 的第 $j$ 个分量为零. 不难证明 $\boldsymbol{J}(i,j,\theta)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{J}(i,j,\theta)=\boldsymbol{D}$. 这样, 若有一个重复的 $d_i$, 我们就能使 $\boldsymbol{z}$ 的一个分量为零. (b) 若 $z_i=0, z_j\neq 0$, 且 $i<j$, 则令 $\boldsymbol{P}$ 是 $i$ 列和 $j$ 列互换的单位矩阵, 可推出 $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{P}$ 是对角矩阵, $(\boldsymbol{P}^{\mathrm{T}}\boldsymbol{z})_i\neq 0$ 且 $(\boldsymbol{P}^{\mathrm{T}}\boldsymbol{z})_j=0$. 这样, 我们就能把为零的 $z_i$ 放在“底部”. 很明显, 重复 (a) 和 (b) 就能得到所需的标准结构. $\boldsymbol{V}_1$ 是这些旋转矩阵之积. □

关于我们上面所给出的求解算法的讨论, 可见 Barlow(1993) 以及该文的参考文献.

### 8.5.4　分而治之方法

我们现在来讨论计算三对角矩阵 $\boldsymbol{T}$ 的 Schur 分解

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{Q}=\boldsymbol{\Lambda}=\operatorname{diag}(\lambda_1,\cdots,\lambda_n),\quad \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}=\boldsymbol{I} \tag{8.5.5}$$

的分而治之方法, 它包括 (a)"撕"$\boldsymbol{T}$ 为两半, (b) 计算这两部分的 Schur 分解, (c) 合并两个半尺寸的 Schur 分解为所需的全尺寸的 Schur 分解. 整个算法由 Dongarra and Sorensen(1987) 提出, 它适于并行计算.

我们首先证明 $\boldsymbol{T}$ 在秩 1 修正下怎样被"分成"两半. 为简单起见, 设 $n=2m$. 定义 $\boldsymbol{v}\in\mathbb{R}^n$ 如下

$$v=\begin{bmatrix} e_m^{(m)} \\ \theta e_1^{(m)} \end{bmatrix}. \tag{8.5.6}$$

注意到对所有的 $\rho\in\mathbb{R}$, 除了"中间四个"元素

$$\widetilde{\boldsymbol{T}}(m:m+1,m:m+1)=\begin{bmatrix} a_m-\rho & b_m-\rho\theta \\ b_m-\rho\theta & a_{m+1}-\rho\theta^2 \end{bmatrix}$$

外, 矩阵 $\widetilde{\boldsymbol{T}}=\boldsymbol{T}-\rho\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$ 和 $\boldsymbol{T}$ 相等. 若我们令 $\rho\theta=b_m$, 则

$$\boldsymbol{T}=\begin{bmatrix} \boldsymbol{T}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{T}_2 \end{bmatrix}+\rho\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}},$$

其中

$$\boldsymbol{T}_1\begin{bmatrix} a_1 & b_1 & & \cdots & 0 \\ b_1 & a_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & b_{m-1} \\ 0 & \cdots & & b_{m-1} & \widetilde{a}_m \end{bmatrix},\quad \boldsymbol{T}_2=\begin{bmatrix} \widetilde{a}_{m+1} & b_{m+1} & & \cdots & 0 \\ b_{m+1} & a_{m+2} & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & & b_{n-1} & a_n \end{bmatrix}$$

且 $\widetilde{a}_m=a_m-\rho,\widetilde{a}_{m+1}=a_{m+1}-\rho\theta^2$.

现在假设我们有 $m\times m$ 正交矩阵 $\boldsymbol{Q}_1$ 和 $\boldsymbol{Q}_2$ 使得 $\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{T}_1\boldsymbol{Q}_1=\boldsymbol{D}_1$ 和 $\boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{T}_2\boldsymbol{Q}_2=\boldsymbol{D}_2$ 均为对角矩阵, 若我们令

$$\boldsymbol{U}=\begin{bmatrix} \boldsymbol{Q}_1 & 0 \\ 0 & \boldsymbol{Q}_2 \end{bmatrix},$$

则

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{U}=\boldsymbol{U}^{\mathrm{T}}\left(\begin{bmatrix} \boldsymbol{T}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{T}_2 \end{bmatrix}+\rho\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}\right)\boldsymbol{U}=\boldsymbol{D}+\rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}},$$

其中

$$\boldsymbol{D}=\begin{bmatrix} \boldsymbol{D}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_2 \end{bmatrix}$$

是对角矩阵且

$$\boldsymbol{z}=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{v}=\begin{bmatrix}\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{e}_m\\ \theta\boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{e}_1\end{bmatrix}.$$

比较这些等式我们看到, 有效地综合这两个半尺寸的 Schur 分解需快速而稳定地计算一个正交矩阵 $\boldsymbol{V}$ 使得

$$\boldsymbol{V}^{\mathrm{T}}(\boldsymbol{D}+\rho\boldsymbol{z}\boldsymbol{z}^{\mathrm{T}})\boldsymbol{V}=\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_1,\cdots,\lambda_n),$$

对此我们在 8.5.3 节中已作了讨论.

### 8.5.5 并行实现

已作了撕开和综合运算后, 我们准备演示整个过程以及在多个处理器上是怎样实现的. 为清楚见, 设 $n=8N$, $N$ 为正整数, 可作三级撕开. 像图 8.5.1 所示, 我们用二叉树来图示此过程. 指标标注在二叉树上. 图 8.5.2 描画了一个结点, 它代表三对角矩阵 $T(b)$ 的特征系统从三对角矩阵 $\boldsymbol{T}(b_0)$ 和 $\boldsymbol{T}(b_1)$ 的特征系统所获得. 例如, $N\times N$ 矩阵 $\boldsymbol{T}(110)$ 和 $\boldsymbol{T}(111)$ 的特征系统综合在一起就产生了 $2N\times 2N$ 三对角矩阵 $\boldsymbol{T}(11)$ 的特征系统.

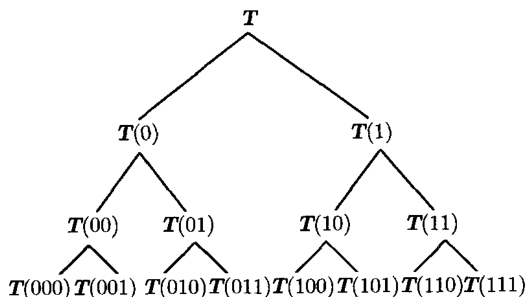



图 8.5.1 计算树

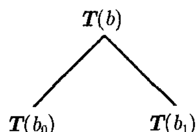



图 8.5.2 一个结点的综合

用树结构算法, 总会存在危险, 当树"爬"到根部时, 并行性会丢失, 但在我们的问题中不会出现这一情形. 为看清这一点, 我们仍假定有 8 个处理器, 且 Proc($b$) 的第一个任务是计算 $\boldsymbol{T}(b)$ 的 Schur 分解, 其中 $b=000,001,010,011,100,101,110,111$, 这部分计算达到最佳装载平衡并且不涉及处理器之间的通信. (我们忽视定理 8.5.4 的降阶, 这不大可能导致明显的装载不平衡.)

在下一个级上, 有四个连在一起的操作需完成: $\boldsymbol{T}(00),\boldsymbol{T}(01),\boldsymbol{T}(10),\boldsymbol{T}(11)$. 然而, 这些计算都可很好地细分, 于是我们分两个处理器给每个任务. 例如, 一旦 $\boldsymbol{T}(00)$ 的特征方程被 Proc(000) 和 Proc(001) 所知, 则它们都得到一半的特征值和相关特征向量. 类似地, 4 个处理器分配给 $\boldsymbol{T}(0)$ 和 $\boldsymbol{T}(1)$ 问题. 所有 8 个处理器都参与计算 $\boldsymbol{T}$ 的特征值. 这样, 在每级水平上, 完全并行性能保持, 因为特征值/特征向量计算是互相独立的.

## 习　题

**8.5.1** 设 $\lambda$ 是对称三对角矩阵 $\boldsymbol{T}$ 的特征值. 证明若 $\lambda$ 有代数重数 $k$, 则 $\boldsymbol{T}$ 的次对角元素至少有 $k-1$ 个为零.

**8.5.2** 试给出一个确定 (8.5.6) 中 $\rho$ 和 $\theta$ 的算法, 具有性质 $\theta \in \{-1, 1\}$ 且 $\min\{|a_r - \rho|, |a_{r+1} - \rho|\}$被极大化.

**8.5.3** 令 $p_r(\lambda) = \det(T(1:r, 1:r) - \lambda \boldsymbol{I}_r)$, 其中 $\boldsymbol{T}$ 由 (8.5.1) 给出. 试给出计算 $p_n'(\lambda)$ 的迭代式, 并用它来导出能计算 $T$ 的特征值的牛顿迭代式.

**8.5.4** 在分配给特殊的 $T_b$ 上的处理器之间, 什么通信是必须的? 处理重复 $d_i$ 以及零元素 $z_i$ 的工作可以分享吗?

**8.5.5** 若 $\boldsymbol{T}$ 为正定矩阵, 可以推出 8.5.4 节中的矩阵 $\boldsymbol{T}_1$ 和 $\boldsymbol{T}_2$ 是正定矩阵吗?

**8.5.6** 设

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{v} \\ \boldsymbol{v}^{\mathrm{T}} & d_{nn} \end{bmatrix},$$

其中 $\boldsymbol{D} = \mathrm{diag}(d_1, \cdots, d_{n-1})$ 有不同的对角元且 $\boldsymbol{v} \in \mathbb{R}^{n-1}$ 的元素都不为零. (a) 证明: 若 $\lambda \in \lambda(\boldsymbol{A})$, 则 $\boldsymbol{D} - \lambda \boldsymbol{I}_{n-1}$ 非奇. (b) 证明: 若 $\lambda \in \lambda(\boldsymbol{A})$, 则 $\lambda$ 是

$$f(\lambda) = \lambda + \sum_{k=1}^{n-1} \frac{v_k^2}{d_k - \lambda} - d_n$$

的零点.

**8.5.7** 设 $\boldsymbol{A} = \boldsymbol{S} + \sigma \boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}$, 其中 $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ 是反对称矩阵, $\boldsymbol{u} \in \mathbb{R}^n, \sigma \in \mathbb{R}$, 说明怎样计算一个正交矩阵 $\boldsymbol{Q}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T} + \sigma \boldsymbol{e}_1\boldsymbol{e}_1^{\mathrm{T}}$, 其中 $\boldsymbol{T}$ 是三对角矩阵且反对称, $\boldsymbol{e}_1$ 是 $\boldsymbol{I}_n$ 的第一列.

**8.5.8** 已知 $\lambda \in \lambda(\boldsymbol{T})$, 其中 $\boldsymbol{T} \in \mathbb{R}^{n\times n}$ 为没有零次对角元的对称三对角矩阵. 说明怎样从方程 $\boldsymbol{T}\boldsymbol{x} = \lambda \boldsymbol{x}$ 计算 $x(1:n-1)$, 给定 $x_n = 1$.

## 本节注释与参考文献

二分法和 Strum 序列方法的讨论见:

W. Barth, R. S. Martin, and J. H. Wilkinson (1967). "Calculation of the Eigenvalues of a Symmetric Tridiagonal Matrix by the Method of Bisection," *Numer. Math.* 9, 386–393. 也见 Wilkinson and Reinsch (1971, 249–256).

K. K. Gupta (1972). "Solution of Eigenvalue Problems by Sturm Sequence Method," *Int. J. Numer. Meth. Eng.* 4, 379–404.

本节所讨论的分而治之算法的许多方面之细节见下列文章:

G. H. Golub (1973). "Some Modified Matrix Eigenvalue Problems," *SIAM Review* 15, 318–344.

J. R. Bunch, C. P. Nielsen, and D. C. Sorensen (1978). "Rank-One Modification of the Symmetric Eigenproblem," *Numer. Math.* 31, 31–48.

J. J. M. Cuppen (1981). "A Divide and Conquer Method for the Symmetric Eigenproblem," *Numer. Math.* 36, 177–195.

J. J. Dongarra and D. C. Sorensen (1987). "A Fully Parallel Algorithm for the Symmetric Eigenvalue Problem," *SIAM J. Sci. and Stat. Comp.* 8, s139–s154.

S. Crivelli and E. R. Jessup (1995). "The Cost of Eigenvalue Computation on Distributed Memory MIMD Computers," *Parallel Computing* 21, 401–422.

该方法所要求的很详细的计算在下列文章中仔细分析了：

J. L. Barlow (1993). "Error Analysis of Update Methods for the Symmetric Eigenvalue Problem," *SIAM J. Matrix Anal. Appl.* 14, 598–618.

带状对称特征值问题的各种推广已被探讨：

P. Arbenz, W. Gander, and G. H. Golub (1988). "Restricted Rank Modification of the Symmetric Eigenvalue Problem: Theoretical Considerations," *Lin. Alg. and Its Applic.* 104, 75–95.

P. Arbenz and G. H. Golub (1988). "On the Spectral Decomposition of Hermitian Matrices Subject to Indefinite Low Rank Perturbations with Applications," *SIAM J. Matrix Anal. Appl.* 9, 40–58.

一个基于"箭头"矩阵 (见题 8.5.7) 的相应分而治之方法在下文中给出：

M. Gu and S. C. Eisenstat (1995). "A Divide-and-Conquer Algorithm for the Symmetric Tridiagonal Eigenproblem," *SIAM J. Matrix Anal. Appl.* 16, 172–191.

## 8.6 计算 SVD

矩阵 $A$ 的奇异值分解与对称矩阵 $A^TA, AA^T$ 及 $\begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$ 的 Schur 分解之间有着重要的关系. 事实上, 如果

$$U^TAV = \mathrm{diag}(\sigma_1, \cdots, \sigma_n)$$

是 $A \in \mathbb{R}^{m\times n}(m \geqslant n)$ 的 SVD, 则

$$V^T(A^TA)V = \mathrm{diag}(\sigma_1^2, \cdots, \sigma_n^2) \in \mathbb{R}^{n\times n}, \tag{8.6.1}$$

$$U^T(AA^T)U = \mathrm{diag}(\sigma_1^2, \cdots, \sigma_n^2, \underbrace{0, \cdots, 0}_{m-n}) \in \mathbb{R}^{m\times m}. \tag{8.6.2}$$

而且, 若

$$U = [\underset{n}{U_1} \quad \underset{m-n}{U_2}],$$

且定义正交矩阵 $Q \in \mathbb{R}^{(m+n)\times(m+n)}$ 为

$$Q = \frac{1}{\sqrt{2}}\begin{bmatrix} V & V & 0 \\ U_1 & -U_1 & \sqrt{2}U_2 \end{bmatrix},$$

则

$$Q^T\begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}Q = \mathrm{diag}(\sigma_1, \cdots, \sigma_n, -\sigma_1, \cdots, -\sigma_n, \underbrace{0, \cdots, 0}_{m-n}). \tag{8.6.3}$$

对称特征问题的这些关系使我们可以把前两节中的数学结论及算法结论应用到奇异值问题. 这一节的优秀参考书包括 Lawson and Hauson(1974) 以及 Stewart and Sun(1990).

### 8.6.1 扰动理论和性质

基于 8.1 节的定理, 我们首先对 SVD 建立一些扰动结果. 回忆一下 $\sigma_i(A)$ 表示 $\boldsymbol{A}$ 的第 $i$ 大的奇异值.

**定理 8.6.1** 若 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, 则对 $k=1:\min\{m,n\}$

$$\sigma_k(A)=\max_{\substack{\dim(\boldsymbol{S})=k\\ \dim(\boldsymbol{T})=k}}\ \min_{\substack{\boldsymbol{x}\in S\\ \boldsymbol{y}\in T}}\frac{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2}=\max_{\dim(S)=k}\ \min_{\boldsymbol{x}\in S}\frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2},$$

注意表达式中 $\boldsymbol{S}\subseteq\mathbb{R}^n$ 和 $\boldsymbol{T}\subseteq\mathbb{R}^m$ 为子空间.

**证明** 把定理 8.1.2 应用于 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 就得到最右边的式子, 余下的证明留作练习. □

**推论 8.6.2** 若 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{E}$ 均属于 $\mathbb{R}^{m\times n}, m\geqslant n$, 则对 $k=1:n$ 有

$$|\sigma_k(\boldsymbol{A}+\boldsymbol{E})-\sigma_k(\boldsymbol{A})|\leqslant\sigma_1(\boldsymbol{E})=\|\boldsymbol{E}\|_2.$$

**证明** 将推论 8.1.6 应用于

$$\begin{bmatrix}\mathbf{0} & \boldsymbol{A}^{\mathrm{T}}\\ \boldsymbol{A} & \mathbf{0}\end{bmatrix}\quad 及\quad \begin{bmatrix}\mathbf{0} & (\boldsymbol{A}+\boldsymbol{E})^{\mathrm{T}}\\ \boldsymbol{A}+\boldsymbol{E} & \mathbf{0}\end{bmatrix}.$$ □

**例 8.6.1** 若

$$\boldsymbol{A}=\begin{bmatrix}1&4\\2&5\\3&6\end{bmatrix},\quad \boldsymbol{A}+\boldsymbol{E}=\begin{bmatrix}1&4\\2&5\\3&6.01\end{bmatrix}$$

则 $\sigma(\boldsymbol{A})=\{9.5080,0.7729\}$ 且 $\sigma(\boldsymbol{A}+\boldsymbol{E})=\{9.5145,0.7706\}$. 很清楚, 对 $i=1:2$, 我们有 $|\sigma_i(\boldsymbol{A}+\boldsymbol{E})-\sigma_i(\boldsymbol{A})|\leqslant\|\boldsymbol{E}\|_2=0.01$.

**推论 8.6.3** 令 $\boldsymbol{A}=[\boldsymbol{a}_1,\cdots,\boldsymbol{a}_n]\in\mathbb{R}^{m\times n}$ 按列划分, $m\geqslant n$. 若 $\boldsymbol{A}_r=[\boldsymbol{a}_1,\cdots,\boldsymbol{a}_r]$, 则对 $r=1:n-1$,

$$\sigma_1(\boldsymbol{A}_{r+1})\geqslant\sigma_1(\boldsymbol{A}_r)\geqslant\sigma_2(\boldsymbol{A}_{r+1})\geqslant\cdots\geqslant\sigma_r(\boldsymbol{A}_{r+1})\geqslant\sigma_r(\boldsymbol{A}_r)\geqslant\sigma_{r+1}(\boldsymbol{A}_{r+1}).$$

**证明** 应用推论 8.1.7 于 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$. □

最后这一结果是说矩阵增加一列, 则最大奇异值增加, 最小奇异值减小.

**例 8.6.2**

$$\boldsymbol{A}=\begin{bmatrix}1&6&11\\2&7&12\\3&8&13\\4&9&14\\5&10&15\end{bmatrix}\Rightarrow\begin{cases}\sigma(\boldsymbol{A}_1)=\{7.4162\},\\ \sigma(\boldsymbol{A}_2)=\{19.5377,1.8095\},\\ \sigma(\boldsymbol{A}_3)=\{35.1272,2.4654,0.0000\}.\end{cases}$$

由此可证实推论 8.6.3.

下一个结果是奇异值问题的 Wielandt-Hoffman 定理.

**定理 8.6.4** 若 $\boldsymbol{A}$ 和 $\boldsymbol{A}+\boldsymbol{E}$ 属于 $\mathbb{R}^{m\times n}, m\geqslant n$, 则

$$\sum_{k=1}^{n}(\sigma_k(\boldsymbol{A}+\boldsymbol{E})-\sigma_k(\boldsymbol{A}))^2\leqslant\|\boldsymbol{E}\|_F^2.$$

**证明** 应用定理 8.1.4 于

$$\begin{bmatrix} \mathbf{0} & \boldsymbol{A}^{\mathrm{T}} \\ \boldsymbol{A} & \mathbf{0} \end{bmatrix} \text{ 和 } \begin{bmatrix} \mathbf{0} & (\boldsymbol{A}+\boldsymbol{E})^{\mathrm{T}} \\ \boldsymbol{A}+\boldsymbol{E} & \mathbf{0} \end{bmatrix}.$$ □

**例 8.6.3** 若

$$\boldsymbol{A} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \quad \text{和} \quad \boldsymbol{A}+\boldsymbol{E} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6.01 \end{bmatrix},$$

则 $\sum_{k=1}^{2}(\sigma_k(\boldsymbol{A}+\boldsymbol{E}) - \sigma_k(\boldsymbol{A}))^2 = 0.472 \times 10^{-4} \leqslant 10^{-4} = \|\boldsymbol{E}\|_F^2$. 见例 8.6.1.

对于 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, 我们说 $k$ 维子空间 $\boldsymbol{S} \subseteq \mathbb{R}^n$ 和 $\boldsymbol{T} \subseteq \mathbb{R}^m$ 形成一奇异子空间对, 若 $\boldsymbol{x} \in \boldsymbol{S}$ 和 $\boldsymbol{y} \in \boldsymbol{T}$ 隐含 $\boldsymbol{Ax} \in T$ 和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y} \in S$. 下列结果与奇异子空间对的扰动有关.

**定理 8.6.5** 给定 $\boldsymbol{A}, \boldsymbol{E} \in \mathbb{R}^{m\times n}(m \geqslant n)$, 并假定 $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ 和 $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ 正交. 设

$$\boldsymbol{V} = [\underset{r}{\boldsymbol{V}_1} \quad \underset{n-r}{\boldsymbol{V}_2}], \quad \boldsymbol{U} = [\underset{r}{\boldsymbol{U}_1} \quad \underset{m-r}{\boldsymbol{U}_2}],$$

且$\mathrm{ran}(\boldsymbol{V}_1)$ 和 $\mathrm{ran}(\boldsymbol{U}_1)$ 形成 $\boldsymbol{A}$ 的奇异子空间对, 令

$$\boldsymbol{U}^{\mathrm{H}}\boldsymbol{AV} = \begin{bmatrix} \boldsymbol{A}_{11} & 0 \\ 0 & \boldsymbol{A}_{22} \end{bmatrix} \begin{matrix} r \\ m-r \end{matrix} \quad (\text{列: } r,\ n-r)$$

$$\boldsymbol{U}^{\mathrm{H}}\boldsymbol{EV} = \begin{bmatrix} \boldsymbol{E}_{11} & \boldsymbol{E}_{12} \\ \boldsymbol{E}_{21} & \boldsymbol{E}_{22} \end{bmatrix} \begin{matrix} r \\ m-r \end{matrix} \quad (\text{列: } r,\ n-r)$$

且设

$$\delta = \min_{\substack{\sigma\in\sigma(\boldsymbol{A}_{11}) \\ \gamma\in\sigma(\boldsymbol{A}_{22})}} |\sigma - \gamma| > 0,$$

若

$$\|\boldsymbol{E}\|_F \leqslant \frac{\delta}{4},$$

则存在矩阵 $\boldsymbol{P} \in \mathbb{R}^{(n-r)\times r}$ 且 $\boldsymbol{Q} \in \mathbb{R}^{(m-r)\times r}$ 满足

$$\left\| \begin{bmatrix} \boldsymbol{Q} \\ \boldsymbol{P} \end{bmatrix} \right\|_F \leqslant 4\frac{\|\boldsymbol{E}\|_F}{\delta},$$

使得 $\mathrm{ran}(\boldsymbol{V}_1 + \boldsymbol{V}_2\boldsymbol{Q})$ 和 $\mathrm{ran}(\boldsymbol{U}_1 + \boldsymbol{U}_2\boldsymbol{P})$ 是 $\boldsymbol{A}+\boldsymbol{E}$ 的奇异子空间对.

**证明** 见 Stewart(1973), 定理 6.4. □

粗略地讲, 该定理指出 $\boldsymbol{A}$ 的 $O(\varepsilon)$ 变化能导致奇异子空间 $\varepsilon/\delta$ 量级的改变, 其中 $\delta$ 为相应奇异值分离程度的一个量度.

**例 8.6.4** 矩阵 $\boldsymbol{A} = \mathrm{diag}(2.000, 1.001, 0.999) \in \mathbb{R}^{4\times 3}$ 有奇异子空间对 $(\mathrm{span}\{v_i\}, \mathrm{span}\{u_i\})$, 对 $i = 1, 2, 3$, 其中 $v_i = e_i^{(3)}$ 且 $u_i = e_i^{(4)}$. 设

$$A+E=\begin{bmatrix}2.000 & 0.010 & 0.010\\0.010 & 1.001 & 0.010\\0.010 & 0.010 & 0.999\\0.010 & 0.010 & 0.010\end{bmatrix}$$

矩阵的相应列

$$\hat{U}=[\hat{u}_1\quad\hat{u}_2\quad\hat{u}_3]=\begin{bmatrix}0.9999 & -0.0144 & 0.0007\\0.0101 & 0.7415 & 0.6708\\0.0101 & 0.6707 & -0.7616\\0.0051 & 0.0138 & -0.0007\end{bmatrix}$$

$$\hat{V}=[\hat{v}_1\quad\hat{v}_2\quad\hat{v}_3]=\begin{bmatrix}0.9999 & -0.0143 & 0.0007\\0.0101 & 0.7416 & 0.6708\\0.0101 & 0.6707 & -0.7416\end{bmatrix}$$

定义了 $A+E$ 的奇异子空间对. 注意到, 对 $i=1,\{\text{span}\{\hat{v}_i\}\}$, $\text{span}\{\hat{u}_i\}\}$ 接近于 $\{\text{span}\{v_i\},\text{span}\{u_i\}\}$, 但对 $i=2$ 或 $i=3$, 不是如此. 另一方面, 奇异子空间对 $\{\text{span}\{\hat{v}_2,\hat{v}_3\},\text{span}\{\hat{u}_2,\hat{u}_3\}\}$ 靠近 $\{\text{span}\{v_2,v_3\},\text{span}\{u_2,u_3\}\}$.

### 8.6.2 SVD 算法

现在, 我们指出怎样用 QR 算法的变形来计算矩阵 $A\in\mathbb{R}^{m\times n}$ 的 SVD, $m\geqslant n$. 乍一看, 这十分简单. 等式 (8.6.1) 使我们想到:

- 形成 $C=A^{\mathrm{T}}A$
- 用对称 QR 算法计算 $V_1^{\mathrm{T}}CV_1=\text{diag}(\sigma_i^2)$
- 用带列主元的 QR 算法于 $AV_1$ 求得

$$U^{\mathrm{T}}(AV_1)\varPi=R.$$

从 $R$ 的列已正交化得出 $U^{\mathrm{T}}A(V_1\varPi)$ 是对角矩阵. 但是, 正像在例 5.3.2 所看到的, $A^{\mathrm{T}}A$ 的形成可能要导致信息的损失. 由于用原矩阵 $A$ 计算 $U$, 情况还不那么十分坏.

Golub and Kahan(1965) 叙述了计算 SVD 较好的方法. 他们的技巧在于对 $A^{\mathrm{T}}A$ 用隐式对称 QR 算法, 同时求 $U$ 和 $V$. 第一步用算法 5.4.2 化 $A$ 为上双对角型

$$U_B^{\mathrm{T}}AV_B=\begin{bmatrix}B\\0\end{bmatrix},\quad B=\begin{bmatrix}d_1 & f_1 & & \cdots & 0\\0 & d_2 & \ddots & & \vdots\\ & \ddots & \ddots & \ddots & \\\vdots & & \ddots & \ddots & f_{n-1}\\0 & \cdots & & 0 & d_n\end{bmatrix}\in\mathbb{R}^{n\times n}.$$

于是, 剩下的问题是计算 $B$ 的 SVD. 为此, 考虑对三对角矩阵 $T=B^{\mathrm{T}}B$ 应用隐式位移 QR 算法 (算法 8.3.2):

- 计算矩阵

$$T(m:n,m:n)=\begin{bmatrix} d_m^2+f_{m-1}^2 & d_m f_m \\ d_m f_m & d_n^2+f_m^2 \end{bmatrix}, \quad m=n-1$$

的靠近 $d_n^2+f_m^2$ 的特征值 $\lambda$.

- 计算 $c_1=\cos(\theta_1)$ 和 $s_1=\sin(\theta_1)$ 使得

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} d_1^2-\lambda \\ d_1 f_1 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}.$$

且令 $\boldsymbol{G}_1=\boldsymbol{G}(1,2,\theta_1)$.

- 计算 Givens 旋转 $\boldsymbol{G}_2,\cdots,\boldsymbol{G}_{n-1}$, 使得当 $\boldsymbol{Q}=\boldsymbol{G}_1\cdots\boldsymbol{G}_{n-1}$ 时, $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{Q}$ 为三对角矩阵, 且 $\boldsymbol{Q}e_1=\boldsymbol{G}_1e_1$.

注意到这需要显式计算 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$. 正如我们所看到的, 从数值观点来看, 这是不明智的.

而假设把上面的 Givens 旋转 $\boldsymbol{G}_1$ 直接应用在 $\boldsymbol{B}$ 上, 演示 $n=6$ 时的情形如下:

$$\boldsymbol{B}\longleftarrow\boldsymbol{B}\boldsymbol{G}_1=\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ + & \times & \times & 0 & 0 & 0 \\ 0 & 0 & \times & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

我们然后能够决定 Givens 旋转 $\boldsymbol{U}_1,\boldsymbol{V}_2,\boldsymbol{U}_2,\cdots,\boldsymbol{V}_{n-1},\boldsymbol{U}_{n-1}$, 把不想要的非零元素在双对角矩阵中往下赶:

$$\boldsymbol{B}\longleftarrow\boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{B}=\begin{bmatrix} \times & \times & + & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & \times & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

$$\boldsymbol{B}\longleftarrow\boldsymbol{B}\boldsymbol{V}_2=\begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & + & \times & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

$$B \longleftarrow U_2^{\mathrm{T}} B = \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & + & 0 & 0 \\ 0 & 0 & \times & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

以后类似. 过程结束时得到新的双对角矩阵 $\bar{B}$, 它与 $B$ 的关系如下:

$$\bar{B} = (U_{n-1}^{\mathrm{T}} \cdots U_1^{\mathrm{T}}) B (G_1 V_2 \cdots V_{n-1}) = \bar{U}^{\mathrm{T}} B \bar{V}.$$

从每个 $V_i$ 的形式 $V_i = G(i, i+1, \theta_i)(i = 2 : n-1)$ 得出 $\bar{V} e_1 = Q e_1$. 由隐式 $Q$ 定理可以断言 $\bar{V}$ 和 $Q$ 实质上是相同的. 这样, 通过直接处理双对角矩阵 $B$ 就可隐式地完成从 $T$ 到 $\bar{T} = \bar{B}^{\mathrm{T}} \bar{B}$ 的转变.

当然, 要使结论成立, 所论的三对角矩阵必须不可约. 由于 $B^{\mathrm{T}} B$ 的次对角元的形式为 $d_{i-1} f_i$, 显然, 必须检查双对角带上的零元素, 若对某个 $k$, 有 $f_k = 0$, 则

$$B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} \begin{matrix} k \\ n-k \end{matrix}$$
$$\qquad\quad k \qquad n-k$$

而原来的 SVD 问题就分成关于 $B_1$ 与 $B_2$ 的两个较小的问题. 若对某个 $k < n, d_k$=0, 则左乘一系列 Givens 变换可使 $f_k$ 为零. 例如, 若 $n = 6$ 且 $k = 3$, 则使行平面 (3, 4), (3, 5) 及 (3, 6) 进行旋转, 就可把第 3 行化为零.

$$B = \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix} \xrightarrow{(3,4)} \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & + & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

$$\xrightarrow{(3,5)} \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix} \xrightarrow{(3,6)} \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

若 $d_n = 0$, 则对平面 $(n-1, n), (n-2, n), \cdots, (1, n)$ 进行一系列旋转, 把最后一列化为零. 这样, 若 $f_1 \cdots f_{n-1} = 0$ 或 $d_1 \cdots d_n = 0$, 则我们可以解耦.

**算法 8.6.1 (Golub-Kahan SVD 步骤)** 给定双对角矩阵 $B \in \mathbb{R}^{n \times n}$, 其对角元与超对角元不为零, 下述算法用双对角矩阵 $\bar{B} = \bar{U}^{\mathrm{T}} B V$ 覆盖 $B$, 其中 $\bar{U}$ 和 $\bar{V}$ 是正交矩阵, 实质上, $\bar{V}$ 是对 $T = B^{\mathrm{T}} B$ 应用算法 8.3.2 获得的正交矩阵.

令 $\mu$ 是 $\boldsymbol{T}=\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$ 的尾部 $2\times 2$ 子矩阵靠近 $t_{nn}$ 的特征值

$y=t_{11}-\mu$

$z=t_{12}$

**for** $k=1:n-1$

  确定 $c=\cos(\theta)$ 和 $s=\sin(\theta)$ 使得

$$[y\quad z]\begin{bmatrix} c & s \\ -s & c \end{bmatrix}=[*\quad 0]$$

  $\boldsymbol{B}=\boldsymbol{B}G(k,k+1,\theta)$

  $y=b_{kk}; z=b_{k+1,k}$

  确定 $c=\cos(\theta)$ 和 $s=\sin(\theta)$ 使得

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^{\mathrm{T}}\begin{bmatrix} y \\ z \end{bmatrix}=\begin{bmatrix} * \\ 0 \end{bmatrix}$$

  $\boldsymbol{B}=G(k,k+1,\theta)^{\mathrm{T}}\boldsymbol{B}$

  **if** $k<n-1$

   $y=b_{k,k+1}; z=b_{k,k+2}$

  **end**

**end**

为了有效地实现该算法, 应分别用数组 $a(1:n)$ 和 $f(1:n-1)$ 来存储 $\boldsymbol{B}$ 的对角元和超对角元, 并且需 $30n$ 个 flop 和 $2n$ 个平方根运算. 累积 $\boldsymbol{U}$ 另需 $6mn$ 个 flop, 累积 $\boldsymbol{V}$ 需 $6n^2$ 个 flop.

一般地, 经过几次上述的 SVD 迭代, 超对角元 $f_{n-1}$ 变得微不足道. 判定 $\boldsymbol{B}$ 的带中的元素为微小的准则, 通常为

$$|f_i|\leqslant\varepsilon(|d_i|+|d_{i+1}|)$$

$$|d_i|\leqslant\varepsilon\|\boldsymbol{B}\|$$

其中 $\varepsilon$ 为单位舍入误差的一个小倍数; 而 $\|\cdot\|$ 是计算上简便的范数.

把算法 5.4.2(双对角化)、算法 8.6.1 以及前面所述的分离计算结合起来, 便可得到下述算法.

**算法 8.6.2 (SVD 算法)** 给定 $\boldsymbol{A}\in\mathbb{R}^{m\times n}(m\geqslant n)$ 和 $\varepsilon$(为单位舍入的小倍数), 下述算法计算 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}=\boldsymbol{D}+\boldsymbol{E}$ 来覆盖 $\boldsymbol{A}$, 其中 $\boldsymbol{U}\in\mathbb{R}^{m\times n}$ 为正交矩阵, $\boldsymbol{V}\in\mathbb{R}^{n\times n}$ 为正交矩阵, $\boldsymbol{D}\in\mathbb{R}^{m\times n}$ 为对角矩阵, $\boldsymbol{E}$ 满足 $\|\boldsymbol{E}\|_2\approx\mathbf{u}\|\boldsymbol{A}\|_2$.

用算法 5.4.2 来计算双对角线化.

$$\begin{bmatrix} \boldsymbol{B} \\ 0 \end{bmatrix}\leftarrow(\boldsymbol{U}_1\cdots\boldsymbol{U}_n)^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{V}_1\cdots\boldsymbol{V}_{n-2}).$$

**until** $q=n$.

  对 $i=1:n-1$, 若 $|b_{i,i+1}|\leqslant\varepsilon(|b_{ii}|+|b_{i+1,i+1}|)$, 置 $b_{i,i+1}$ 为零.

找最大 $q$ 和最小 $p$ 使得若

$$
\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_{11} & 0 & 0 \\ 0 & \boldsymbol{B}_{22} & 0 \\ 0 & 0 & \boldsymbol{B}_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}
$$

$$
\begin{matrix} p & n-p-q & q \end{matrix}
$$

则 $\boldsymbol{B}_{33}$ 为对角矩阵且 $\boldsymbol{B}_{22}$ 的所有超对角元不为零.

**if** $q < n$

**if** $B_{22}$ 的某对角元为零, 则调整该行的超对角元为零.

**else**

对 $B_{22}$ 应用算法 8.6.1

$B = \text{diag}(I_p, U, I_{q+m-n})^{\text{T}} B \ \text{diag}(I_p, V, I_q)$

**end**

**end**

**end**

该算法所需工作量及其数值性质已在 5.4.5 节和 5.5.8 节讨论.

**例 8.6.5**　若把算法 8.6.2 应用于

$$
\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 4 \end{bmatrix}
$$

则超对角元如下所示收敛于零:

| 迭　代 | $O(\lvert a_{21}\rvert)$ | $O(\lvert a_{32}\rvert)$ | $O(\lvert a_{43}\rvert)$ |
|---|---|---|---|
| 1 | $10^{0}$ | $10^{0}$ | $10^{0}$ |
| 2 | $10^{0}$ | $10^{0}$ | $10^{0}$ |
| 3 | $10^{0}$ | $10^{0}$ | $10^{0}$ |
| 4 | $10^{0}$ | $10^{-1}$ | $10^{-2}$ |
| 5 | $10^{0}$ | $10^{-1}$ | $10^{-8}$ |
| 6 | $10^{0}$ | $10^{-1}$ | $10^{-27}$ |
| 7 | $10^{0}$ | $10^{-1}$ | 收敛 |
| 8 | $10^{0}$ | $10^{-4}$ | |
| 9 | $10^{-1}$ | $10^{-14}$ | |
| 10 | $10^{-1}$ | 收敛 | |
| 11 | $10^{-4}$ | | |
| 12 | $10^{-12}$ | | |
| 13 | 收敛 | | |

观察到收敛是立方的.

### 8.6.3　Jacobi SVD 算法

将 8.4 节中的 Jacobi 算法用于求解 SVD 问题是很显然的. 代替求解一系列 $2 \times 2$ 对称特征值问题, 我们求解一系列的 $2 \times 2$ SVD 问题. 这样, 对一给定的指标

对 $(p,q)$, 我们计算一对旋转矩阵使得

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{bmatrix} \begin{bmatrix} c_2 & s_2 \\ -s_2 & c_2 \end{bmatrix} = \begin{bmatrix} d_p & 0 \\ 0 & d_q \end{bmatrix}.$$

见习题 8.6.8. 所产生的算法称为**双边的**, 因为每个校正包含一个左乘和右乘.

**单边的**Jacobi 算法包含一系列的成对向量的正交化, 对一给定指标对 $(p,q)$, 选定一个 Jacobi 旋转 $\boldsymbol{J}(p,q,\theta)$ 使得 $\boldsymbol{AJ}(p,q,\theta)$ 的 $p$ 列和 $q$ 列互相正交. 见习题 8.6.8. 注意这对应于使 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 中 $(p,q)$ 元和 $(q,p)$ 元为零. 一旦 $\boldsymbol{AV}$ 的列足够正交, SVD 的其他部分 ($\boldsymbol{U}$ 和 $\boldsymbol{\Sigma}$) 可由列加权 $\boldsymbol{AV}=\boldsymbol{U\Sigma}$ 得出.

## 习 题

**8.6.1** 证明若 $\boldsymbol{B}\in\mathbb{R}^{n\times n}$ 是具有重奇异值的上双对角矩阵, 则 $\boldsymbol{B}$ 在其对角线或超对角线上一定有一个零.

**8.6.2** 用 $\boldsymbol{A}\in\mathbb{R}^{m\times n}(m\geqslant n)$ 的奇异向量来给出 $\begin{bmatrix} 0 & \boldsymbol{A}^{\mathrm{T}} \\ \boldsymbol{A} & 0 \end{bmatrix}$ 的特征向量公式.

**8.6.3** 给出一个用复 Householder 变换来化一个复矩阵 $\boldsymbol{A}$ 为实双对角型的算法.

**8.6.4** 试找出 $\begin{bmatrix} \boldsymbol{B} & -\boldsymbol{C} \\ \boldsymbol{C} & \boldsymbol{B} \end{bmatrix}$ 的奇异值和奇异向量与 $\boldsymbol{A}=\boldsymbol{B}+\mathrm{i}\boldsymbol{C}(\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{m\times n})$ 的奇异值和奇异向量之间的关系.

**8.6.5** 完成定理 8.6.1 的证明.

**8.6.6** 设 $n=2m$ 且 $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ 为反对称的三对角矩阵, 证明存在一个置换矩阵 $\boldsymbol{P}\in\mathbb{R}^{n\times n}$ 使得 $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{SP}$ 具有如下形式:

$$\boldsymbol{P}^{\mathrm{T}}\boldsymbol{SP}=\begin{bmatrix} \boldsymbol{0} & -\boldsymbol{B}^{\mathrm{T}} \\ \boldsymbol{B} & \boldsymbol{0} \end{bmatrix}\begin{matrix} m \\ m \end{matrix}.$$
$$\qquad\qquad m \qquad m$$

描述 $\boldsymbol{B}$. 说明如何利用 $\boldsymbol{B}$ 的 SVD 分解计算 $\boldsymbol{S}$ 的特征值和特征向量. 同样考虑 $n=2m+1$ 的情形.

**8.6.7** (a) 令

$$\boldsymbol{C}=\begin{bmatrix} w & x \\ y & z \end{bmatrix}$$

为实矩阵, 给出一个稳定算法来计算 $c$ 和 $s$, 满足 $c^2+s^2=1$, 使得

$$\boldsymbol{B}=\begin{bmatrix} c & s \\ -s & c \end{bmatrix}\boldsymbol{C}$$

为对称矩阵. (b) 用 Jacobi 三角计算并结合 (a) 给出一稳定算法用以计算 $\boldsymbol{C}$ 的 SVD. (c) (b) 部分可用来给出计算 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 的 SVD 的 Jacobi 型算法. 对于给定的 $(p,q), p<q$, 确定 Jacobi 变换 $\boldsymbol{J}(p,q,\theta_1)$ 和 $\boldsymbol{J}(p,q,\theta_2)$ 使得若

$$\boldsymbol{B}=\boldsymbol{J}(p,q,\theta_1)^{\mathrm{T}}\boldsymbol{AJ}(p,q,\theta_2),$$

则 $b_{pq}=b_{qp}=0$. 证明

$$\mathrm{off}(\boldsymbol{B})^2=\mathrm{off}(\boldsymbol{A})^2-b_{pq}^2-b_{qp}^2.$$

怎样确定 $p, q$ 呢? 此算法应作何修正, 以用来处理当 $A \in \mathbb{R}^{m\times n}(m > n)$ 的情形呢?

**8.6.8** 令 $x, y \in \mathbb{R}^m$, 定义正交矩阵 $Q$ 为

$$Q = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}.$$

给出一计算 $c$ 和 $s$ 的稳定算法, 使得 $[x, y]Q$ 的列相互正交.

**8.6.9** 设 $B \in \mathbb{R}^{n\times n}$ 为上双对角矩阵, $b_{nn} = 0$. 说明如何构造正交矩阵 $U$ 和 $V$(一系列 Givens 变换之乘积), 使得 $U^TBV$ 为第 $n$ 列为零的上双对角矩阵.

**8.6.10** 设 $B \in \mathbb{R}^{n\times n}$ 是具有对角元素 $d(1:n)$ 和超对角元素 $f(1:n-1)$ 的上双对角矩阵. 陈述并证明定理 8.5.1 的奇异值分解.

## 本节注释与参考文献

SVD 的数学性质在 Stewart and Sun(1990) 中讨论到, 也在下列文献中谈到:

A. R. Amir-Moez(1965). *Extremal Properties of Linear Transformations and Geometry of Unitary Spaces,* Texas Tech University Mathematics Series, no. 243, Lubbock, Texas.

G. W. Stewart (1973). "Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems," *SIAM Review* 15, 727–764.

P. A. Wedin (1972). "Perturbation Bounds in Connection with the Singular Value Decomposition," *BIT* 12, 99-111.

G. W. Stewart (1979). "A Note on the Perturbation of Singular Values," *Lin. Alg. and Its Applic.* 28, 213–216.

G. W. Stewart (1984). "A Second Order Perturbation Expansion for Smal Singular Values," *Lin. Alg. and Its Applic.* 56, 231–236.

R. J. Vaccaro (1994). "A Second-Order Perturbation Expansion for the SVD," *SIAM J. Matrix Anal. Applic,* 15, 661–671.

选择对称 QR 算法来计算 SVD 的思想首先出现在:

G. H. Golub and W. Kahan (1965). "Calculating the Singular Values and Pseudo-Inverse of a Matrix," *SIAM J. Num. Anal. Ser.* B2, 205–224.

之后有一些早期的实现:

P. A. Businger and G. H. Golub (1969). "Algorithm 358: Singular Value Decomposition of a Complex Matrix," *Comm. Assoc. Comp. Mach.* 12, 564–565.

G. H. Golub and C. Reinsch (1970). "Singular Value Decomposition and Least Squares Solutions," *Numer. Math.* 14, 403–420. See also Wilkinson and Reinsch (1971, 134–151).

与 SVD 相关的算法进展见于:

J. J. M. Cuppen (1983). "The Singular Value Decomposition in Product Form," *SIAM J. Sci. and Stat. Comp.* 4, 216–222.

J. J. Dongarra (1983). "Improving the Accuracy of Computed Singular Values," *SIAM J. Sci. and Stat. Comp.* 4, 712–719.

S. Van Huffel, J. Vandewalle, and A. Haegemans (1987)."An Efficient and Reliable Algorithm

for Computing the Singular Subspace of a Matrix Associated with its Smallest Singular Values," *J. Comp. and Appl. Math.* 19, 313–330.

P. Deift, J. Demmel, L. -C. Li, and C. Tomei (1991). "The Bidiagonal Singular Value Decomposition and Hamiltonian Mechanics," *SIAM J. Num. Anal.* 28, 1463–1516.

R. Mathias and G. W. Stewart (1993). "A Block QR Algorithm and the Singular Value Decomposition," *Lin. Alg. and Its Applic.* 182. 91–100.

Å. Björck, E. Grimme, and P. Van Dooren (1994). "An Implicit Shift Bidiagonalization Algorithm for Ill-Posed Problems," *BIT* 34, 510–534.

一个矩阵的极分解能通过 SVD 立即求得. 然而, 针对这一目的也有一些特殊算法.

N. J. Higham (1986), "Computing the Polar Decomposition-with Applications," *SIAM J. Sci. and Stat. Comp.* 7, 1160–1174.

N. J. Higham and P. Papadimitriou (1994). "A Parallel Algorithm for Computing the Polar Decomposition," *Parallel Comp.* 20, 1161–1173.

SVD 的 Jacobi 算法归纳为两类. 双边 Jacobi 算法重复进行校正 $A \leftarrow U^{\mathrm{T}}AV$ 产生一列逐渐对角化的迭代序列.

E. G. Kogbetliantz (1955). "Solution of Linear Equations by Diagonalization of Coefficient Matrix," *Quart. Appl. Math.* 13, 123–132.

G. E. Forsythe and P. Henrici (1960). "The Cyclic Jacobi Method for Computing the Principal Values of a Complex Matrix," *Trans. Amer. Math. Soc,* 94, 1–23.

C. C. Paige and P. Van Dooren(1986). "On the Quadratic Convergence of Kogbetliantz's Algorithm for Computing the Singular Value Decomposition," *Lin. Alg. and Its Applic.* 77, 301–313.

J. P. Charlier and P. Van Dooren (1987). "On Kogbetliantz's SVD Algorithm in the Presence of Clusters," *Lin. Alg. and Its Applic.* 95, 135–160.

Z. Bai (1988). "Note on the Quadratic Convergence of Kogbetliantz's Algorithm for Computing the Singular Value Decomposition," *Lin. Alg. and Its Applic.* 104, 131–140.

J. P. Charlier, M. Vanbegin, P. Van Dooren (1988). "On Efficient Implementation of Kogbetliantz's Algorithm for Computing the Singular Value Decomposition," *Numer. Math.* 52, 279–300.

K. V. Fernando (1989). "Linear Convergence of the Row Cyclic Jacobi and Kogbetliantz methods," *Numer. Math.* 56, 73–92.

单边 Jacobi SVD 程序重复做校正 $A \leftarrow AV$ 产生一列逐渐正交化的迭代序列.

J. C. Nash (1975). "A One-Sided Tranformation Method for the Singular Value Decomposition and Algebraic Eigenproblem," *Comp. J.* 18, 74–76.

P. C. Hansen (1988). "Reducing the Number of Sweeps in Hestenes Method," in *Singular Value Decomposition and Signal Processing,* ed. E. F. Deprettere, North Holland.

K. Veselič and V. Hari (1989). "A Note on a One-Sided Jacobi Algorithm," *Numer. Math.* 56, 627–633.

人们也提出了许多并行实现.

F. T. Luk (1980). "Computing the Singular Value Decomposition on the ILLIAC IV," *ACM*

*Trans. Math. Soft.* 6, 524–539.

R. P. Brent and F. T. Luk (1985). "The Solution of Singular Value and Symmetric Eigenvalue Problems on Multiprocessor Arrays," *SIAM J. Sci. and Stat. Comp.* 6, 69–84.

R. P. Brent, F. T. Luk, and C. Van Loan (1985). "Computation of the Singular Value Decomposition Using Mesh Connected Processors," *J. VLSI Computer Systems* 1, 242–270.

F. T. Luk (1986). "A Triangular Processor Array for Computing Singular Values," *Lin. Alg. and Its Applic.* 77, 259–274.

M. Berry and A. Sameh (1986). "Multiprocessor Jacobi Algorithms for Dense Symmetric Eigenvalue and Singular Value Decompositions," in *Proc. International Conference on Parallel Processing,* 433–440.

R. Schreiber (1986). "Solving Eigenvalue and Singular Value Problems on an Undersized Systolic Array," *SIAM J. Sci. and Stat. Comp.* 7, 441–451.

C. H. Bischof and C. Van Loan (1986). "Computing the SVD on a Ring of Array Processors," in *Large Scale Eigenvalue Problems,* eds. J. Cullum and R. Willoughby, North Holland, 51–66.

C. H. Bischof (1987). "The Two-Sided Block Jacobi Method on Hypercube Architectures," in *Hypercube Multiprocessors.* ed. M. T. Heath. SIAM Press, Philadelphia.

C. H. Bischof (1989). "Computing the Singular Value Decomposition on a Distributed System of Vector Processors," *Parallel Computing* 11, 171–186.

S. Van Huffel and H. Park (1994). "Parallel Tri-and Bidiagonalization of Bordered Bidiagonal Matrices," *Parallel Computing* 20, 1107-1128.

B. Lang (1996). "Parallel Reduction of Banded Matrices to Bidiagonal Form," *Parallel Computing* 22, 1–18.

专门为对称特征问题设计的分而治之的算法可推广到 SVD 算法：

E. R. Jessup and D. C. Sorensen (1994). "A Parallel Algorithm for Computing the Singular Value Decomposition of a Matrix," *SIAM J. Matrix Anal. Appl.* 15, 530–548.

M. Gu and S. C. Eisenstat (1995). "A Divide-and-Conquer Algorithm for the Bidiagonal SVD," *SIAM J. Matrix Anal. Appl.* 16, 79–92.

对 SVD 计算的仔细分析包括：

J. W. Demmel and W. Kahan (1990). "Accurate Singular Values of Bidiagonal Matrices," *SIAM J. Sci. and Stat. Comp.* 11, 873–912.

K. V. Fernando and B. N. Parlett (1944). "Accurate Singular Values and Differential qd Algorithms," *Numer. Math.* 67, 191–230.

S. Chandrasekaren and I. C. F. Ipsen (1994). "Backward Errors for Eigenvalue and Singular Value Decompositions," *Numer. Math.* 68, 215–223.

高精度的 SVD 计算以及与 Cholesky 计算、Schur 计算和奇异值计算讨论于下述文章中：

J. W. Demmel and K. Veselič(1992). "Jacobi's Method is More Accurate than QR," *SIAM J. Matrix Anal. Appl.* 13, 1204–1245.

R. Mathias (1995). "Accurate Eigensystem Computations by Jacobi Methods," *SIAM J.*

*Matrix Anal. Appl.* 16, 977–1003.

## 8.7 一些广义特征值问题

给定一对称矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 和一对称正定矩阵 $\boldsymbol{B}\in\mathbb{R}^{n\times n}$, 我们考虑如何寻找一非零向量 $x$ 和标量 $\lambda$ 使得 $\boldsymbol{A}x=\lambda\boldsymbol{B}x$. 这就是*对称正定广义特征问题*. 标量 $\lambda$ 被认为是*广义特征值*. 随着 $\lambda$ 变动, $\boldsymbol{A}-\lambda\boldsymbol{B}$ 定义了一*矩阵束*, 我们的任务是确定

$$\lambda(\boldsymbol{A},\boldsymbol{B})=\{\lambda|\det(\boldsymbol{A}-\lambda\boldsymbol{B})=0\}.$$

一个对称正定广义特征问题用一个同余变换可以化为一个等价的问题:

$$\boldsymbol{A}-\lambda\boldsymbol{B}\text{奇异}\Leftrightarrow(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X})-\lambda(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X})\text{奇异}.$$

这样, 若 $\boldsymbol{X}$ 非奇异, 则 $\lambda(\boldsymbol{A},\boldsymbol{B})=\lambda(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X},\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X})$.

在这一节, 我们要介绍通过仔细选择 $\boldsymbol{X}$ 来提出求解这种特征问题的保持结构不变的各种算法. 有关的广义奇异值分解也同时讨论.

### 8.7.1 数学背景

我们要找一个稳定而有效的算法来计算 $\boldsymbol{X}$, 使得 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 和 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X}$ 都化为“标准型”. 所追求的明显形式是对角型.

**定理 8.7.1** *设 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 是 $n\times n$ 对称矩阵, 定义 $\boldsymbol{C}(\mu)$ 为*

$$\boldsymbol{C}(\mu)=\mu\boldsymbol{A}+(1-\mu)\boldsymbol{B},\quad \mu\in\mathbb{R}. \tag{8.7.1}$$

*若存在一 $\mu\in[0,1]$, 使得 $\boldsymbol{C}(\mu)$ 是非负定矩阵, 并且*

$$\mathrm{null}(\boldsymbol{C}(\mu))=\mathrm{null}(\boldsymbol{A})\cap\mathrm{null}(\boldsymbol{B}),$$

*则存在非奇异矩阵 $\boldsymbol{X}$, 使得 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 和 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X}$ 都是对角矩阵.*

**证明** 选取 $\mu\in[0,1]$, 使得 $\boldsymbol{C}(\mu)$ 为非负定矩阵且具有性质 $\mathrm{null}(\boldsymbol{C}(\mu))=\mathrm{null}(\boldsymbol{A})\cap\mathrm{null}(\boldsymbol{B})$. 令

$$\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{C}(\mu)\boldsymbol{Q}_1=\begin{bmatrix}\boldsymbol{D}&0\\0&0\end{bmatrix},\quad \boldsymbol{D}=\mathrm{diag}(d_1,\cdots,d_k),\quad d_i>0$$

是 $\boldsymbol{C}(\mu)$ 的 Schur 分解并定义 $\boldsymbol{X}_1=\boldsymbol{Q}_1\mathrm{diag}(\boldsymbol{D}^{-1/2},\boldsymbol{I}_{n-k})$. 若 $\boldsymbol{A}_1=\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}_1,\boldsymbol{B}_1=\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X}_1$, 且 $\boldsymbol{C}_1=\boldsymbol{X}_1^{\mathrm{T}}\boldsymbol{C}(\mu)\boldsymbol{X}_1$, 则

$$\boldsymbol{C}_1=\begin{bmatrix}\boldsymbol{I}_k&0\\0&0\end{bmatrix}=\mu\boldsymbol{A}_1+(1-\mu)\boldsymbol{B}_1.$$

由于 $\mathrm{span}\{e_{k+1},\cdots,e_n\}=\mathrm{null}(\boldsymbol{C}_1)=\mathrm{null}(\boldsymbol{A}_1)\cap\mathrm{null}(\boldsymbol{B}_1)$, 所以 $\boldsymbol{A}_1$ 和 $\boldsymbol{B}_1$ 都有以下分块结构:

$$\boldsymbol{A}_1=\begin{bmatrix}\boldsymbol{A}_{11}&0\\0&0\end{bmatrix}\begin{matrix}k\\n-k\end{matrix}\qquad \boldsymbol{B}_1=\begin{bmatrix}\boldsymbol{B}_{11}&0\\0&0\end{bmatrix}\begin{matrix}k\\n-k\end{matrix}$$
$$\qquad k\quad n-k\qquad\qquad\qquad k\quad n-k$$

而且 $\boldsymbol{I}_k = \mu\boldsymbol{A}_{11} + (1-\mu)\boldsymbol{B}_{11}$.

设 $\mu \neq 0$. 可知若 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{B}_{11}\boldsymbol{Z} = \mathrm{diag}(b_1, \cdots, b_k)$ 是 $\boldsymbol{B}_{11}$ 的 Schur 分解且我们令 $\boldsymbol{X} = \boldsymbol{X}_1\mathrm{diag}(\boldsymbol{Z}, \boldsymbol{I}_{n-k})$, 则

$$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X} = \mathrm{diag}(b_1, \cdots, b_k, 0, \cdots, 0) \equiv \boldsymbol{D}_B,$$

$$\begin{aligned}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X} &= \frac{1}{\mu}\boldsymbol{X}^{\mathrm{T}}(\boldsymbol{C}(\mu) - (1-\mu)\boldsymbol{B})\boldsymbol{X} \\ &= \frac{1}{\mu}\left(\begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} - (1-\mu)\boldsymbol{D}_B\right) \equiv \boldsymbol{D}_A.\end{aligned}$$

另一方面, 若 $\mu = 0$, 则令 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{A}_{11}\boldsymbol{Z} = \mathrm{diag}(a_1, \cdots, a_k)$ 是 $\boldsymbol{A}_{11}$ 的 Schur 分解, 且令 $\boldsymbol{X} = \boldsymbol{X}_1\mathrm{diag}(\boldsymbol{Z}, \boldsymbol{I}_{n-k})$. 易证, 在这种情形下, $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 和 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X}$ 为对角矩阵. □

通常, 由于 $\boldsymbol{A}$ 或者 $\boldsymbol{B}$ 是正定矩阵, 定理 8.7.1 的条件能够满足.

**推论 8.7.2** 若 $\boldsymbol{A} - \lambda\boldsymbol{B} \in \mathbb{R}^{n\times n}$ 是对称正定的, 则存在一非奇异矩阵 $\boldsymbol{X} = [x_1, \cdots, x_n]$ 使得

$$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X} = \mathrm{diag}(a_1, \cdots, a_n), \quad \boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X} = \mathrm{diag}(b_1, \cdots, b_n),$$

而且, 对 $i = 1:n$, $\boldsymbol{A}x_i = \lambda_i\boldsymbol{B}x_i$, 其中 $\lambda_i = a_i/b_i$.

**证明** 在定理 8.7.1 中令 $\mu = 0$ 我们看到对称正定束同时对角化. 推论的其余部分易证明. □

**例 8.7.1** 若

$$\boldsymbol{A} = \begin{bmatrix} 229 & 163 \\ 163 & 116 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 81 & 59 \\ 59 & 43 \end{bmatrix},$$

则 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 是对称正定矩阵且 $\lambda(\boldsymbol{A}, \boldsymbol{B}) = \{5, -1/2\}$. 若

$$\boldsymbol{X} = \begin{bmatrix} 3 & -5 \\ -4 & 7 \end{bmatrix},$$

则 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X} = \mathrm{diag}(5, -1)$ 及 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{X} = \mathrm{diag}(1, 2)$.

Stewart(1979) 给出满足

$$c(\boldsymbol{A}, \boldsymbol{B}) = \min_{\|\boldsymbol{x}\|_2=1} (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})^2 + (\boldsymbol{x}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{x})^2 > 0 \tag{8.7.2}$$

的对称束 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 的扰动理论. 标量 $c(\boldsymbol{A}, \boldsymbol{B})$ 称作束 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 的 Crawford数.

**定理 8.7.3** 设 $\boldsymbol{A} - \lambda\boldsymbol{B}$ 是一个 $n \times n$ 对称正定束, 具有特征值

$$\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n.$$

设 $\boldsymbol{E}_A$ 和 $\boldsymbol{E}_B$ 是 $n \times n$ 对称矩阵, 且满足

$$\varepsilon^2 = \|\boldsymbol{E}_A\|_2^2 + \|\boldsymbol{E}_B\|_2^2 < c(\boldsymbol{A}, \boldsymbol{B}).$$

则 $(\boldsymbol{A} + \boldsymbol{E}_A) - \lambda(\boldsymbol{B} + \boldsymbol{E}_B)$ 对称正定, 其特征值

$$\mu_1 \geqslant \cdots \geqslant \mu_n,$$

对 $i = 1:n$ 满足

$$|\arctan(\lambda_i) - \arctan(\mu_i)| \leqslant \arctan(\varepsilon/c(\boldsymbol{A}, \boldsymbol{B})).$$

**证明** 见 Stewart(1979). □

### 8.7.2 求对称正定问题的方法

我们转来考虑算法, 首先给出一个既用 Cholesky 分解又用对称 QR 算法的求解对称正定问题的方法.

**算法 8.7.1** 给定 $A = A^{\mathrm{T}} \in \mathbb{R}^{n\times n}$ 和 $B = B^{\mathrm{T}} \in \mathbb{R}^{n\times n}$, 其中 $B$ 为正定矩阵, 本算法计算一非奇异矩阵 $X$ 使得 $X^{\mathrm{T}}BX = I_n$ 且 $X^{\mathrm{T}}AX = \mathrm{diag}(a_1, \cdots, a_n)$.

用算法 4.2.2 计算 Cholesky 分解 $B = GG^{\mathrm{T}}$.

计算 $C = G^{-1}AG^{-\mathrm{T}}$.

用对称 QR 算法计算 Schur 分解 $Q^{\mathrm{T}}CQ = \mathrm{diag}(a_1, \cdots, a_n)$.

令 $X = G^{-\mathrm{T}}Q$.

此算法需约 $14n^3$ 个 flop. 在实际实现中, $A$ 能被矩阵 $C$ 覆盖. 详见 Martin and Wilkinson (1968c). 注意到

$$\lambda(A, B) = \lambda(A, GG^{\mathrm{T}}) = \lambda(G^{-1}AG^{-\mathrm{T}}, I) = \lambda(C) = \{a_1, \cdots, a_n\}.$$

若 $\hat{a}_i$ 是由算法 8.7.1 求得的特征值, 则能证明 $\hat{a}_i \in \lambda(G^{-1}AG^{-\mathrm{T}} + E_i)$, 其中 $\|E_i\|_2 \approx \mathbf{u}\|A\|_2\|B^{-1}\|_2$. 这样, 若 $B$ 是病态的, 则由于舍入误差影响, 即使 $a_i$ 是良态的广义特征值, 也会严重损害 $\hat{a}_i$ 的值. 当然, 在此情况下, 矩阵 $C = G^{-1}AG^{-\mathrm{T}}$ 可能有一些很大的元素, 故若 $B$ 是病态的, 则 $G$ 也是病态的. 在算法 8.7.1 中用 $VD^{-1/2}$ 代替矩阵 $G$, 有时能克服这一困难, 其中 $V^{\mathrm{T}}BV = D$ 是 $B$ 的 Schur 分解. 若 $D$ 的对角元从小到大排序, 则 $C$ 的大元集中在左上角. 这样, 计算 $C$ 的一些小特征值不会有太大的舍入误差影响 (或者说直观上应如此). 更进一步的讨论, 请参阅 Wilkinson(1965, 337~338 页).

**例 8.7.2** 若

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}, \quad G = \begin{bmatrix} 0.001 & 0 & 0 \\ 1 & 0.001 & 0 \\ 2 & 1 & 0.001 \end{bmatrix}$$

及 $B = GG^{\mathrm{T}}$, 则 $A - \lambda B$ 的两个小特征值为

$$a_1 = -0.619\,402\,940\,600\,584, \quad a_2 = 1.627\,440\,079\,051\,887.$$

若使用 17 位浮点运算, 则把对称 QR 算法用到 $fl(D^{-1/2}V^{\mathrm{T}}AVD^{-1/2})$ 时, 计算的特征值达到机器精度, 其中 $B = VDV^{\mathrm{T}}$ 是 $B$ 的 Schur 分解. 另一方面, 若用算法 8.7.1, 则

$$\hat{a}_1 = -0.619\,373\,517\,376\,444, \quad \hat{a}_2 = 1.627\,516\,601\,905\,228,$$

只得到 4 位准确有效数字, 其原因在于 $\kappa_2(B) \approx 10^{18}$.

用 $A$ 和 $B$ 的适当凸组合代替 $B$, 有时能改善算法 8.7.1 中矩阵 $X$ 的条件. 变动后的矩阵束之特征值与原矩阵束特征值的关系已在定理 8.7.1 的证明中详细说明.

与算法 8.7.1 有关的其他困难都是围绕如下事实：即使 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是稀疏矩阵，$\boldsymbol{G}^{-1}\boldsymbol{A}\boldsymbol{G}^{-\mathrm{T}}$ 也是满的. 这是一个严重问题，因为实际中许多对称正定问题都是大型稀疏的.

Crawford(1973) 已指明当 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 为带状矩阵时，如何实现算法 8.7.1. 但是，除此之外，同时对角化方法对大型稀疏对称-正定问题是行不通的.

另一个想法是将 Rayleigh 商迭代 (8.4.4) 作如下推广：

给定 $x_0$, $\|x_0\|_2 = 1$

**for** $k = 0, 1, \cdots$

$$\mu_k = x_k^{\mathrm{T}} A x_k / x_k^{\mathrm{T}} B x_k \tag{8.7.3}$$

　　解 $(A - \mu_k B) z_{k+1} = B x_k$ 得 $z_{k+1}$

　　$x_{k+1} = z_{k+1} / \|z_{k+1}\|_2$

**end**

该迭代的数学基础是

$$\lambda = \frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{x}} \tag{8.7.4}$$

极小化

$$f(\lambda) = \|\boldsymbol{A}\boldsymbol{x} - \lambda \boldsymbol{B}\boldsymbol{x}\|_B, \tag{8.7.5}$$

其中 $\|\cdot\|_B$ 定义为 $\|\boldsymbol{z}\|_B^2 = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{B}^{-1}\boldsymbol{z}$. (8.7.3) 与 (8.4.4) 有相似的数学性质，其适用性取决于能否容易地求解形如 $(\boldsymbol{A} - \lambda\boldsymbol{B})\boldsymbol{z} = \boldsymbol{x}$ 的方程组. 类似的评论可用于下列广义正交迭代：

给定 $\boldsymbol{Q}_0 \in \mathbb{R}^{n\times p}$, 且 $Q_0^{\mathrm{T}} Q_0 = I_p$

**for** $k = 1, 2, \cdots$

$$\text{从 } BZ_k = AQ_{k-1} \text{ 解出 } Z_k \tag{8.7.6}$$

　　$Z_k = Q_k R_k$　(QR 分解)

**end**

在数学上，这等价于在 (7.3.4) 中用 $\boldsymbol{B}^{-1}\boldsymbol{A}$ 替换 $\boldsymbol{A}$. 其实用性与解取决于能否容易地求解形如 $\boldsymbol{B}\boldsymbol{z} = \boldsymbol{y}$ 的线性方程组.

$\boldsymbol{A}$ 和 $\boldsymbol{B}$ 有时会非常大，以致无法实行 (8.7.3) 或 (8.7.6). 在这种情况下，可以借助于一些梯度算法和一些坐标松弛算法. 丰富的文献介绍可见 Stewart(1976).

### 8.7.3　广义奇异值问题

我们以形如 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} - \lambda\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$(其中 $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{B} \in \mathbb{R}^{p\times n}$) 的对称束的一些评述来结束本节. 这个束构成*广义奇异值分解*(GSVD) 的基础，该分解在一些带约束的最小二乘问题中是有用的. (参阅 12.1 节). 注意到，由定理 8.7.1 可知，存在非奇异矩阵 $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ 使得 $\boldsymbol{X}^{\mathrm{T}}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{X}$ 及 $\boldsymbol{X}^{\mathrm{T}}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})\boldsymbol{X}$ 均为对角矩阵. GSVD 的价值在于不必形成 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 和 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$ 就可完成其对角化.

**定理 8.7.4 (广义奇异值分解)** 若我们有 $A \in \mathbb{R}^{m\times n}, m \geqslant n$ 且 $B \in \mathbb{R}^{p\times n}$, 则存在正交矩阵 $U \in \mathbb{R}^{m\times m}, V \in \mathbb{R}^{p\times p}$ 和可逆矩阵 $X \in \mathbb{R}^{n\times n}$, 使得

$$U^{\mathrm{T}}AX = C = \mathrm{diag}(c_1, \cdots, c_n), \quad c_i \geqslant 0,$$
$$V^{\mathrm{T}}BX = S = \mathrm{diag}(s_1, \cdots, s_q), \quad s_i \geqslant 0,$$

其中 $q = \min(p, n)$.

**证明** 该分解之证明发表于 Van Loan(1976). 我们根据 Paige and Saunders (1981) 给出一个更有构造性的证明. 为清楚起见, 我们设 $\mathrm{null}(A) \cap \mathrm{null}(B) = \{0\}$ 且 $p \geqslant n$. 我们把此证明推广到所有情形的工作留给读者.

令

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R$$

是 QR 分解, 其中 $Q_1 \in \mathbb{R}^{m\times n}, Q_2 \in \mathbb{R}^{p\times n}$, $R \in \mathbb{R}^{n\times n}$. Paige 和 Saunders 证明了在

$$Q_1 = UCW^{\mathrm{T}}, \quad Q_2 = VSW^{\mathrm{T}} \tag{8.7.7}$$

意义下, $Q_1$ 和 $Q_2$ 的奇异值分解是相关的. 这里 $U, V$ 和 $W$ 为正交矩阵,

$$C = \mathrm{diag}(c_i), \quad 0 \leqslant c_1 \leqslant \cdots \leqslant c_n,$$
$$S = \mathrm{diag}(s_i), \quad s_1 \geqslant \cdots \geqslant s_n,$$

且 $C^{\mathrm{T}}C + S^{\mathrm{T}}S = I_n$. (8.7.7) 的分解是 2.6 节的 CS 分解的变形, 从它我们有结论 $A = Q_1R = UC(W^{\mathrm{T}}R)$ 和 $B = Q_2R = VS(W^{\mathrm{T}}R)$.

通过令 $X = (W^{\mathrm{T}}R)^{-1}, D_A = C$ 和 $D_B = S$ 即可得到定理. $R$ 的可逆性由假定 $\mathrm{null}(A) \cap \mathrm{null}(B) = \{0\}$ 可得. □

集合 $\sigma(A, B) \equiv \{c_1/s_1, \cdots, c_n/s_q\}$ 的元素称为 $A$ 和 $B$ 的*广义奇异值*. 注意 $\sigma \in \sigma(A, B)$ 隐含着 $\sigma^2 \in \lambda(A^{\mathrm{T}}A, B^{\mathrm{T}}B)$. 以上定理是 SVD 的推广, 因为若 $B = I_n$, 则 $\sigma(A, B) = \sigma(A)$.

由于 Stewart(1983) 和 Van Loan(1985) 已指出怎样稳定计算 CS 分解, GSVD 的证明在实际上就显得很重要. 唯一棘手部分是求 $W^{\mathrm{T}}R$ 的逆以得到 $X$. 注意到 $X = [x_1, \cdots, x_n]$ 的列满足

$$s_i^2 A^{\mathrm{T}}Ax_i = c_i^2 B^{\mathrm{T}}Bx_i, \quad i = 1:n,$$

这样若 $s_i \neq 0$, 则 $A^{\mathrm{T}}Ax_i = \sigma_i^2 B^{\mathrm{T}}Bx_i$, 其中 $\sigma_i = c_i/s_i$. 于是, $x_i$ 称为对 $(A, B)$ 的*广义奇异向量*.

在几种应用里, 需求某些指定的广义奇异向量子空间 $\mathrm{span}[x_{i_1}, \cdots, x_{i_k}]$ 的一组正交基. 我们来说明不用求任何矩阵的逆或叉积就可完成这个任务.

- 计算 QR 分解

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R.$$

- 计算 CS 分解

$$Q_1 = UCW^{\mathrm{T}} \qquad Q_2 = VSW^{\mathrm{T}}$$

对 $C, S$ 的对角元排序使得

$$\{c_1/s_1, \cdots, c_k/s_k\} = \{c_{i_1}/s_{i_1}, \cdots, c_{i_k}/s_{i_k}\}.$$

- 计算正交矩阵 $Z$ 和上三角形矩阵 $T$ 使得 $TZ = W^{\mathrm{T}}R$(见习题 8.7.5). 注意若 $X^{-1} = W^{\mathrm{T}}R = TZ$, 则 $X = Z^{\mathrm{T}}T^{-1}$ 且 $Z$ 的前 $k$ 行是 $\mathrm{span}\{x_1, \cdots, x_k\}$ 的正交基.

## 习 题

**8.7.1** 设 $A \in \mathbb{R}^{n\times n}$ 是对称矩阵, $G \in \mathbb{R}^{n\times n}$ 是非奇异下三角形矩阵. 给出计算 $C = G^{-1}AG^{-\mathrm{T}}$ 的一个有效算法.

**8.7.2** 设 $A \in \mathbb{R}^{n\times n}$ 对称及 $B \in \mathbb{R}^{n\times n}$ 对称正定. 试给出一个利用 Cholesky 分解和对称 QR 算法计算 $AB$ 特征值的算法.

**8.7.3** 证明如果 $C$ 是实的和可对角化的, 那么存在对称矩阵 $A$ 和 $B$, $B$ 非奇异, 使得 $C = AB^{-1}$. 这证明了对称束 $A - \lambda B$ 基本上是具有一般性的.

**8.7.4** 如果 $A$ 和 $B$ 均为对称和非负定的, 试证明如何将 $Ax = \lambda Bx$ 问题转化为一个广义的奇异值问题.

**8.7.5** 给定 $Y \in \mathbb{R}^{n\times n}$, 证明怎样计算 Householder 矩阵 $H_2, \cdots, H_n$ 使得 $YH_n \cdots H_2 = T$ 是上三角形矩阵. 提示: $H_k$ 将第 $k$ 行化为零.

**8.7.6** 假设

$$\begin{bmatrix} 0 & A \\ A^{\mathrm{T}} & 0 \end{bmatrix}\begin{bmatrix} y \\ z \end{bmatrix} = \lambda \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}\begin{bmatrix} y \\ z \end{bmatrix},$$

其中 $A \in \mathbb{R}^{m\times n}, B_1 \in \mathbb{R}^{m\times m}$, $B_2 \in \mathbb{R}^{n\times n}$. 假定 $B_1$ 和 $B_2$ 是正定的且有相应的 Cholesky 三角形矩阵 $G_1$ 和 $G_2$. 找出该问题的广义特征值和 $G_1^{-1}AG_2^{-\mathrm{T}}$ 的奇异值的关系.

**8.7.7** 设 $A$ 和 $B$ 均为对称正定矩阵. 说明如何用 Cholesky 分解和 CS 分解来计算 $\lambda(A, B)$ 和相应的特征向量.

## 本节注释与参考文献

关于对称正定束的计算方法之极好的综述可见:

G. W. Stewart (1976). "A Bibliographical Tour of the Large Sparse Generalized Eigenvalue Problem," in *Sparse Matrix Computations,* ed., J. R. Bunch and D. J. Rose, Academic Press, New York.

一些特别重要的文章有:

R. S. Martin and J. H. Wilkinson (1968c). "Reduction of a Symmetric Eigenproblem $Ax=\lambda Bx$ and Related Problems to Standard Form," *Numer. Math.* 11, 99–110.

G. Peter and J. H. Wilkinson (1969). "Eigenvalues of $Ax = \lambda Bx$ with Band Symmetric A and B," *Comp. J.* 12 , 398–404.

G. Fix and R. Heiberger (1972). "An Algorithm for the Ill-Conditioned Generalized Eigenvalue Problem," *SIAMJ. Num. Anal.* 9, 78–88.

C. R. Crawford (1973). "Reduction of a Band Symmetric Generalized Eigenvalue Problem," *Comm. ACM* 16, 41–44.

A. Ruhe (1974). "SOR Methods for the Eigenvalue Problem with Large Sparse Matrices," *Math. Comp.* 28, 695–710.

C. R. Crawford (1976). "A Stable Generalized Eigenvalue Problem," *SIAM J. Num. Anal.* 13, 854–860.

A. Bunse-Gerstner (1984). "An Algorithm for the Symmetric Generalized Eigenvalue Problem," *Lin. Alg. and Its Applic.* 58, 43–68.

C. R. Crawford(1986). "Algorithm 646 PDFIND: A Routine to Find a Positive Definite Linear Combination of Two Real Symmetric Matrices," *ACM Trans. Math. Soft.* 12, 278–282.

C. R. Crawford and Y. S. Moon (1983). "Finding a Positive Definite Linear Combination of Two Hermitian Matrices," *Lin. Alg. and Its Applic.* 51, 37–48.

W. Shougen and Z. Shuqin (1991). "An Algorithm for $Ax = \lambda Bx$ with Symmetric and Positive Definite $A$ and $B$," *SIAMJ. Matrix Anal. Appl.* 12, 654–660.

K. Li and T-Y. Li (1993). "A Homotopy Algorithm for a Symmetric Generalized Eigenproblem," *Numerical Algorithms* 4, 167–195.

K. Li, T-Y. Li, and Z. Zeng (1994), "An Algorithm for the Generalized Symmetric Tridiagonal Eigenvalue Problem," *Numerical Algorithms* 8, 269–291.

H. Zhang and W. F. Moss (1994). "Using Parallel Banded Linear System Solvers in Generalized Eigenvalue Problems," *Paraller Computing* 20, 1089–1106.

同时化两个对称矩阵为对角型的方法在下述文中讨论到：

A. Berman and A. Ben-Israel (1971). "A Note on Pencils of Hermitian or Symmetric Matrices," *SIAM J. Applic. Math,* 21, 51–54.

F. Uhlig (1973)."Simultaneous Block Diagonalization of Two Real Symmetric Matrices," *Lin. Alg. and Its Applic.* 7, 281–289.

F. Uhlig (1976). "A Canonical Form for a Pair of Real Symmetric Matrices That Generate a Nonsingular Pencil," *Lin. Alg. and Its Applic.* 14, 189–210.

K. N. Majinder (1979). "Linear Combinations of Hermitian and Real Symmetric Matrices," *Lin. Alg. and Its Applic.* 25, 95–105.

关于对称 - 定型问题我们所提出的扰动理论来源于：

G. W. Stewart (1979). "Perturbation Bounds for the Definite Generalized Eigenvalue Problem," *Lin. Alg. and Its Applic.* 23, 69–86.

也见：

L. Elsner and J. Guang Sun (1982). "Perturbation Theorems for the Generalized Eigenvalue Problem," *Lin. Alg. and its Applic.* 48, 341–357.

J. Guang Sun (1982). "A Note on Stewart's Theorem for Definite Matrix Pairs," *Lin. Alg. and Its Applic.* 48, 331–339.

J. Guang Sun (1983). "Perturbation Analysis for the Generalized Singular Value Problem," *SIAM J. Numer. Anal.* 20, 611–625.

C. C. Paige (1984). "A Note on a Result of Sun J. -Guang: Sensitivity of the CS and GSV Decompositions," *SIAM J. Numer. Anal.* 21, 186–191.

广义 SVD 和一些应用讨论见：

C. F. Van Loan (1976). "Generalizing the Singular Value Decomposition," *SIAM J. Num. Anal.* 13, 76–83.

C. C. Paige and M. Saunders (1981). "Towards A Generalized Singular Value Decomposition," *SIAM J. Num. Anal.* 18, 398–405.

B. Kågström (1985)."The Generalized Singular Value Decomposition and the General $A-\lambda B$ Problem," *BIT* 24, 568–583.

计算 CS 和广义奇异值分解的稳定理论描述见：

G. W. Stewart (1983). "A Method for Computing the Generalized Singular Value Decomposition," in *Matrix Pencils*, ed. B. Kågström and A. Ruhe, Springer-Verlag. New York, pp. 207–220.

C. F. Van Loan (1985). "Computing the CS and Generalized Singular Value Decomposition," *Numer. Math.* 46, 479–492.

M. T. Heath, A. J. Laub, C. C. Paige, and R. C. Ward (1886). "Computing the SVD of a Product of Two Matrices," *SIAM J. Sci. and Stat. Comp.* 7, 1147–1159.

C. C. Paige (1986). "Computing the Generalized Singular Value Decomposition," *SIAM J. Sci. and Stat. Comp.* 7, 1126–1146.

L. M. Ewerbring and F. T. Luk (1989). "Canonical Correlations and Generalized SVD: Applications and New Algorithms," *J. Comput. Appl. Math.* 27, 37–52.

J. Erxiong (1990)."An Algorithm for Finding Generalized Eigenpairs of a Symmetric Definite Matrix Pencil," *Lin. Alg. and Its Applic.* 132, 65–91.

P. C. Hansen (1990). "Relations Between SVD and GSVD of Discrete Regularization Problems in Standard and General Form," *Lin. Alg. and Its Applic.* 141, 165–176.

H. Zha (1991). "The Restricted Singular Value Decomposition of Matrix Triplets," *SIAM J. Matrix Anal. Appl.* 12, 172–194.

B. De Moor and G. H. Golub (1991)."The Restricted Singular Value Decomposition: Properties and Applications," *SIAM J. Matrix Anal. Appl.* 12, 401–425.

V. Hari (1991). "On Paris of Almost Diagonal Matrices," *Lin. Alg. and Its Applic.* 148, 193–223.

B. De Moor and P. Van Dooren (1992). "Generalizing the Singular Value and QR Decompositions," *SIAM J. Matrix Anal. Appl.* 13, 993–1014.

H. Zha (1992). "A Numerical Algorithm for Computing the Restricted Singular Value Decomposition of Matrix Triplets," *Lin. Alg. and Its Applic.* 168, 1–25.

R-C. Li (1993). "Bounds on Perturbations of Generalized Singular Values and of Associated Subspaces," *SIAM J. Matrix Anal. Appl.* 14, 195–234.

K. Veselič (1993). "A Jacobi Eigenreduction Algorithm for Definite Matrix Pairs," *Numer. Math.* 64, 241–268.

Z. Bai and H. Zha (1993). "A New Preprocessing Algorithm for the Computation of the

Generalized Singular Value Decomposition," *SIAM J. Sci. Comp.* 14, 1007–1012.

L. Kaufman (1993). "An Algorithm for the Banded Symmetric Generalized Matrix Eigenvalue Problem," *SIAM J. Matrix Anal. Appl.* 14, 372–389.

G. E. Adams, A. W. Bojanczyk, and F. T. Luk (1994). "Computing the PSVD of Two 2×2 Triangular Matrices," *SIAM J. Matrix Anal. Appl.* 15, 366–382.

Z. Drmač (1994). *The Generalized Singular Value Problem*, Ph. D. Thesis, Fern Universitat, Hagen, Germany.

R-C. Li (1994). "On Eigenvalue Varations of Rayleigh Quotient Matrix Pencils of a Definite Pencil," *Lin. Alg. and Its Applic.* 208/209, 471–483.

# 第 9 章　Lanczos 方法

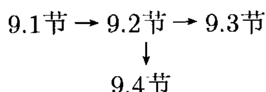
在本章, 我们介绍 Lanczos 方法, 它可以用来求解特定的大规模稀疏对称特征值问题 $Ax = \lambda x$. 该方法涉及对给定的矩阵 $A$ 进行局部三对角化. 然而, 与 Householder 方法不同的是, 在算法过程中不会有满的子矩阵产生. 同样重要的是, $A$ 的两端的特征值的信息早在三对角化完成之前就已出现. 因此, 在只需要矩阵 $A$ 的少量最大或最小特征值的时候, Lanczos 算法有明显的优越性.

在 9.1 节中, 我们导出了该算法并给出了它确切的运算步骤; 详细讨论了 Kaniel-Paige 理论的关键部分. 此理论解释了 Lanczos 方法惊人的收敛性质. 不幸的是, 舍入误差使得 Lanczos 方法在实际中难以应用. 其核心问题是迭代过程中产生的 Lanczos 向量会失去正交性. 为此, 在 9.2 节中, 给出了几种处理这一问题的办法.

在 9.3 节中, 我们将说明如何用 Lanczos 方法的思想来解决奇异值问题、最小二乘问题及线性方程组问题. 特别值得注意的是导出了求解对称正定线性方程组的共轭梯度法. 下一章将进一步探讨 Lanczos 方法与共轭梯度法之间的联系. 在 9.4 节中, 我们讨论了建立在 Hessenberg 分解基础上的 Arnoldi 迭代法和 (有时) 可用来三对角化非对称阵的 Lanczos 方法.

**预备知识**

9.1 节 ~9.3 节需要第 5 章和第 8 章的知识, 9.4 节需要第 7 章的知识. 本章各节间的关系如下:




Brown, Chu, Ellison, and Plemmons(1994) 收集了大量的有关 Lanczos 方法的论文. 其他的补充参考文献包括 Parlett(1980), Saad(1992), Chatelin(1993). Cullum and Willoughby(1985a, 1985b) 的两卷书包括了算法的分析与软件.

## 9.1　方法的导出及收敛性

假定 $A \in \mathbb{R}^{n\times n}$ 为大型、稀疏的对称矩阵, 且只需要求出它的少数几个最大或最小的特征值. 这个问题可用 Lanczos(1950) 中提出的方法解决. 此方法产生一列三对角矩阵 $T \in \mathbb{R}_k^{n\times n}$, 其最大与最小特征值越来越好地近似 $A$ 的最大与最小特征值. 本节将导出这一技巧并研究其确切的运算特性. 在本节中, 用 $\lambda_i$ 代表第 $i$ 个

最大的特征值.

### 9.1.1 Krylov 子空间

导出 Lanczos 算法的方式有好几种. 我们倾向于利用求 Rayleigh 商

$$r(\boldsymbol{x})=\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}},\quad \boldsymbol{x}\neq\mathbf{0}$$

的优化问题来引入该技巧, 从而它惊人的收敛性不会来得太突然. 由定理 8.1.2 知, $r(x)$ 的最大值与最小值分别为 $\lambda_1(\boldsymbol{A})$ 与 $\lambda_n(\boldsymbol{A})$. 假设 $\{q_i\}\subseteq\mathbb{R}^n$ 是正交向量. 令标量 $M_k$ 和 $m_k$ 分别为

$$M_k=\lambda_1(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)=\max_{y\neq 0}\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)\boldsymbol{y}}{y^{\mathrm{T}}y}=\max_{\|y\|_2=1}r(\boldsymbol{Q}_k\boldsymbol{y})\leqslant\lambda_1(\boldsymbol{A}),$$

$$m_k=\lambda_k(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)=\min_{\boldsymbol{y}\neq\mathbf{0}}\frac{\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}=\min_{\|y\|_2=1}r(Q_k\boldsymbol{y})\geqslant\lambda_n(\boldsymbol{A}),$$

其中 $\boldsymbol{Q}_k=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k]$. 考虑如何产生 $\boldsymbol{q}_k$, 使得 $M_k$ 和 $m_k$ 是 $\lambda_1(\boldsymbol{A})$ 和 $\lambda_n(\boldsymbol{A})$ 越来越好的近似, 就可导出 Lanczos 算法.

设 $u_k\in\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}$ 且 $M_k=r(\boldsymbol{u}_k)$. 由于 $r(\boldsymbol{x})$ 在梯度方向

$$\nabla r(\boldsymbol{x})=\frac{2}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}(\boldsymbol{A}\boldsymbol{x}-r(\boldsymbol{x})\boldsymbol{x})$$

增加最快. 所以, 如果选取 $\boldsymbol{q}_{k+1}$ 满足

$$\nabla r(\boldsymbol{u}_k)\in\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_{k+1}\},\tag{9.1.1}$$

我们有 $M_{k+1}>M_k$(这里假定 $\nabla_r(\boldsymbol{u}_k)\neq 0$). 同理, 如果 $\boldsymbol{v}_k\in\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}$ 且 $r(\boldsymbol{v}_k)=m_k$, 我们选取 $q_{k+1}$ 使得

$$\nabla r(\boldsymbol{v}_k)\in\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_{k+1}\},\tag{9.1.2}$$

因为 $r(\boldsymbol{x})$ 在负梯度方向 $-\nabla_r(\boldsymbol{x})$ 下降最快.

初看起来, 要寻找单个向量 $\boldsymbol{q}_{k+1}$ 满足两个条件似乎不可能. 然而, 因为 $\nabla r(\boldsymbol{x})\in\mathrm{span}\{\boldsymbol{x},\boldsymbol{A}\boldsymbol{x}\}$, 如果

$$\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}=\mathrm{span}\{\boldsymbol{q}_1,\boldsymbol{A}\boldsymbol{q}_1,\cdots,\boldsymbol{A}^{k-1}\boldsymbol{q}_1\},$$

且选取 $\boldsymbol{q}_{k+1}$ 使得

$$\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_{k+1}\}=\mathrm{span}\{\boldsymbol{q}_1,\boldsymbol{A}\boldsymbol{q}_1,\cdots,\boldsymbol{A}^{k-1}\boldsymbol{q}_1,\boldsymbol{A}^k\boldsymbol{q}_1\},$$

则很明显 (9.1.1) 和 (9.1.2) 可同时满足. 于是, 导致了计算 Krylov 子空间

$$K(\boldsymbol{A},\boldsymbol{q}_1,k)=\mathrm{span}\{\boldsymbol{q}_1,\boldsymbol{A}\boldsymbol{q}_1,\cdots,\boldsymbol{A}^{k-1}\boldsymbol{q}_1\}$$

的标准正交基问题. 这恰好就是 8.3.2 节中碰到的 Krylov 矩阵

$$K(\boldsymbol{A},\boldsymbol{q}_1,n)=[\boldsymbol{q}_1,\boldsymbol{A}\boldsymbol{q}_1,\boldsymbol{A}^2\boldsymbol{q}_1,\cdots,\boldsymbol{A}^{n-1}\boldsymbol{q}_1]$$

的象空间.

### 9.1.2 三对角化

为了能有效地找出这组基, 我们利用矩阵 $\boldsymbol{A}$ 的三对角化与 $K(\boldsymbol{A},\boldsymbol{q}_1,n)$ 的 QR 分解之间的关系. 设 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 是三对角矩阵, 且 $\boldsymbol{Q}\boldsymbol{e}_1=\boldsymbol{q}_1$, 则

$$K(\boldsymbol{A},\boldsymbol{q}_1,n)=\boldsymbol{Q}[\boldsymbol{e}_1,\boldsymbol{T}\boldsymbol{e}_1,\boldsymbol{T}^2\boldsymbol{e}_1,\cdots,\boldsymbol{T}^{n-1}\boldsymbol{e}_1].$$

这正是 $K(\boldsymbol{A},\boldsymbol{q}_1,n)$ 的 QR 分解, 其中 $\boldsymbol{e}_1=I_n(:,1)$. 因此, 用第一列为 $\boldsymbol{q}_1$ 的正交矩阵来三对角化矩阵 $\boldsymbol{A}$ 就能有效地来求出 $q_k$.

在 8.3.1 节中讨论过的 Householder 三对角化可用来达到这一目的. 然而, 当 $\boldsymbol{A}$ 是大型稀疏矩阵时, 这一方法是不可取的, 因为 Householder 相似变换会破坏矩阵的稀疏性. 其结果是在约化的过程中将产生了无法接受的大型稠密矩阵.

有时, 用 Givens 变换可以控制矩阵稀疏性的破坏, 见 Duff and Reid(1976). 然而, 当 $\boldsymbol{A}$ 稀疏时, 大多数情况下, 任何通过逐步修正矩阵 $\boldsymbol{A}$ 来计算 $\boldsymbol{T}$ 的作法都是不可取的.

这建议我们直接计算三对角矩阵 $\boldsymbol{T}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}$ 的元素. 令 $\boldsymbol{Q}=[\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_n]$,

$$\boldsymbol{T}=\begin{bmatrix}\alpha_1 & \beta_1 & & \cdots & 0\\ \beta_1 & \alpha_2 & \ddots & & \vdots\\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{n-1}\\ 0 & \cdots & & \beta_{n-1} & \alpha_n\end{bmatrix}$$

比较矩阵方程 $\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{Q}\boldsymbol{T}$ 两边的列, 对 $k=1:n-1$ 有

$$\boldsymbol{A}\boldsymbol{q}_k=\beta_{k-1}\boldsymbol{q}_{k-1}+\alpha_k\boldsymbol{q}_k+\beta_k\boldsymbol{q}_{k+1},\quad \beta_0\boldsymbol{q}_0\equiv\boldsymbol{0}.$$

利用 $\{\boldsymbol{q}_i\}$ 的正交性可得 $\alpha_k=\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_k$. 而且, 如果 $r_k=(\boldsymbol{A}-\alpha_k\boldsymbol{I})\boldsymbol{q}_k-\beta_{k-1}\boldsymbol{q}_{k-1}$ 非零, 则 $\boldsymbol{q}_{k+1}=\boldsymbol{r}_k/\beta_k$, 其中 $\beta_k=\pm\|\boldsymbol{r}_k\|_2$. 若 $\boldsymbol{r}_k=\boldsymbol{0}$, 则迭代停止, 但 (我们将看到) 已得到关于不变子空间的有价值的信息. 因此, 整理上述迭代公式的顺序, 我们得到如下的 Lanczos**迭代算法**:

$r_0=q_1;\beta_0=1;q_0=0;k=0$

**while**$(\beta_k\neq 0)$,

$q_{k+1}=r_k/\beta_k;\ k=k+1;\ \alpha_k=q_k^{\mathrm{T}}Aq_k;$ (9.1.3)

$r_k=(A-\alpha_kI)q_k-\beta_{k-1}q_{k-1};\ \beta_k=\|r_k\|_2;$

**end**

不失一般性, 在上述算法中取 $\beta_k$ 为正数. $q_k$ 称为 Lanczos **向量**.

### 9.1.3 终止与误差界

当 $q_1$ 包含在一个真不变子空间中时，在完全三对角化之前迭代就会终止. 这是 Lanczos 方法的几个数学性质之一. 我们将其概括如下.

**定理 9.1.1** *设 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 为对称矩阵, $\boldsymbol{q}_1\in\mathbb{R}^n$ 且 $\|\boldsymbol{q}_1\|_2=1$. 则 Lanczos 迭代 (9.1.3) 进行到第 $k=m$ 步终止, 其中 $m=\text{rank}(K(\boldsymbol{A},\boldsymbol{q}_1,n))$. 此外, 对所有的 $k=1:m$, 有*

$$\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{Q}_k\boldsymbol{T}_k+\boldsymbol{r}_k\boldsymbol{e}_k^{\mathrm{T}}, \tag{9.1.4}$$

*其中*

$$\boldsymbol{T}_k=\begin{bmatrix}\alpha_1 & \beta_1 & & \cdots & 0\\ \beta_1 & \alpha_2 & \ddots & & \vdots\\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{k-1}\\ 0 & \cdots & & \beta_{k-1} & \alpha_k\end{bmatrix}$$

*且 $\boldsymbol{Q}_k=[\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_k]$ 的列正交, $\text{ran}(\boldsymbol{Q}_k)=K(\boldsymbol{A},\boldsymbol{q}_1,k)$.*

**证明** 通过对 $k$ 进行归纳法证明. 假设迭代已产生 $\boldsymbol{Q}_k=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k]$, 且 $\text{ran}(\boldsymbol{Q}_k)=K(\boldsymbol{A},\boldsymbol{q}_1,k)$, $\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{Q}_k=\boldsymbol{I}_k$. 从 (9.1.3) 很容易得知 (9.1.4) 式成立. 因此, $\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{T}_k+\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{r}_k\boldsymbol{e}_k^{\mathrm{T}}$. 既然有 $\alpha_i=\boldsymbol{q}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_i, i=1:k$, 以及

$$\begin{aligned}\boldsymbol{q}_{i+1}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_i&=\boldsymbol{q}_{i+1}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{q}_i-\alpha_i\boldsymbol{q}_i-\beta_{i-1}\boldsymbol{q}_{i-1})\\&=\boldsymbol{q}_{i+1}^{\mathrm{T}}(\beta_i\boldsymbol{q}_{i+1})=\beta_i, \qquad i=1:k-1\end{aligned}$$

成立, 因而 $\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{T}_k$. 所以 $\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{r}_k=\boldsymbol{0}$.

如果 $\boldsymbol{r}_k\neq\boldsymbol{0}$, 则 $\boldsymbol{q}_{k+1}=\boldsymbol{r}_k/\|\boldsymbol{r}_k\|_2$ 与 $\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k$ 正交, 并且

$$\boldsymbol{q}_{k+1}\in\text{span}\{\boldsymbol{A}\boldsymbol{q}_k,\boldsymbol{q}_k,\boldsymbol{q}_{k-1}\}\subseteq K(\boldsymbol{A},\boldsymbol{q}_1,k+1),$$

因此, $\boldsymbol{Q}_{k+1}^{\mathrm{T}}\boldsymbol{Q}_{k+1}=\boldsymbol{I}_{k+1}$, 且 $\text{ran}(\boldsymbol{Q}_{k+1})=K(\boldsymbol{A},\boldsymbol{q}_1,k+1)$. 另一方面, 若 $\boldsymbol{r}_k=\boldsymbol{0}$, 则 $\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{Q}_k\boldsymbol{T}_k$. 这表明 $\text{ran}(\boldsymbol{Q}_k)=K(\boldsymbol{A},\boldsymbol{q}_1,k)$ 为不变子空间. 由此可得 $k=m=\text{rank}(K(\boldsymbol{A},\boldsymbol{q}_1,n))$. □

在 Lanezos 迭代中, $\beta_k$ 为零是受欢迎的, 因为它标志着已计算出一精确的不变子空间. 然而在实际计算中, 恰好出现一个零, 甚至很小的 $\beta_k$, 是很少发生的, 不过, $\boldsymbol{T}_k$ 的最大、最小特征值倒是 $\boldsymbol{A}$ 的最大、最小特征值极好的近似. 因此, 必须寻找关于 $\boldsymbol{T}_k$ 的特征值收敛性的其他解释. 下面就是这样一个结论.

**定理 9.1.2** *假定 Lanczos 算法已进行了 $k$ 步, 且三对角矩阵 $\boldsymbol{T}_k$ 的 Schur 分解为 $\boldsymbol{S}_k^{\mathrm{T}}\boldsymbol{T}_k\boldsymbol{S}_k=\text{diag}(\theta_1,\cdots,\theta_k)$. 令 $\boldsymbol{Y}_k=[y_1,\cdots,y_k]=\boldsymbol{Q}_k\boldsymbol{S}_k\in\mathbb{R}^{n\times k}$, 则*

$$\|\boldsymbol{A}y_i-\theta_iy_i\|_2=|\beta_k|\cdot|s_{ki}|, \quad i=1:k,$$

其中 $S_k = (s_{pq})$.

**证明** 将 (9.1.4) 右乘 $S_k$ 得到

$$AY_k = Y_k\mathrm{diag}(\theta_1, \cdots, \theta_k) + r_k e_k^T S_k,$$

于是

$$Ay_i = \theta_i y_i + r_k(e_k^T S e_i).$$

取范数且记住 $\|r_k\|_2 = |\beta_k|$ 即知定理成立. □

此定理给出了 $T_k$ 与 $A$ 的特征值的一个可计算的误差界:

$$\min_{\mu\in\lambda(A)} |\theta_i - \mu| \leqslant |\beta_k||s_{ki}|, \quad i = 1 : k.$$

注意, 用定理 8.1.15 中的术语, $(\theta_i, y_i)$ 是子空间 $\mathrm{ran}(Q_k)$ 的 Ritz 对.

Golub(1974) 描述了用 $T_k$ 来估计 $A$ 的特征值的另一种方法, 它涉及构造一个秩 1 矩阵 $E$ 使得 $\mathrm{ran}(Q_k)$ 为 $A+E$ 的不变子空间. 特别地, 如果我们用 Lanczos 方法来计算 $AQ_k = Q_kT_k + r_ke_k^T$, 且令 $E = \tau ww^T$, 其中 $\tau = \pm 1$, $w = aq_k + br_k$, 则

$$(A+E)Q_k = Q_k(T_k + \tau a^2 e_k e_k^T) + (1+\tau ab)r_k e_k^T.$$

若 $1 + \tau ab = 0$, 则三对角矩阵 $\widetilde{T}_k = T_k + \tau a^2 e_k e_k^T$ 的特征值也就是 $A+E$ 的特征值. 由定理 8.1.8 知, 对 $i = 2 : k$, 区间 $[\lambda_i(\widetilde{T}_k), \lambda_{i-1}(\widetilde{T}_k)]$ 包含了 $A$ 的一个特征值.

这些区间依赖于 $\tau a^2$ 的选取. 假设已有 $A$ 的一个近似特征值 $\lambda$. $\tau a^2$ 的一种可能的选取是使得 $\det(T_k - \lambda I_k) = (\alpha_2 + \tau a^2 - \lambda)p_{k-1}(\lambda) - \beta_{k-1}^2 p_{k-2}(\lambda) = 0$, 其中 $p_i(x) = \det(T_i - xI_i)$ 可用三项迭代公式 (8.5.2) 计算. 这里假定 $p_{k-1}(\lambda) \neq 0$. Lehmann(1963) 和 Householder(1968) 讨论了这种思想的特征值估计.

### 9.1.4 Kaniel-Paige 收敛性理论

前面的讨论指出如何通过 Lanczos 算法估计特征值, 但它没揭露任何有关收敛速度的信息. 这方面的结果构成了 Kaniel-Paige 理论. 以下就是其中之一.

**定理 9.1.3** *设 $A$ 为 $n\times n$ 对称矩阵, 且特征值为 $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n$, 相应的规范正交特征向量为 $z_1, z_2, \cdots, z_n$. $T_k$ 为用 Lanczos 方法得到的三对角矩阵 (第 $k$ 步得到), 其相应的特征值为 $\theta_1 \geqslant \theta_2 \geqslant \cdots \geqslant \theta_k$, 则*

$$\lambda_1 \geqslant \theta_1 \geqslant \lambda_1 - \frac{(\lambda_1 - \lambda_n)\tan(\phi_1)^2}{(c_{k-1}(1+2\rho_1))^2},$$

*其中 $\cos(\phi_1) = |q_1^T z_1|, \rho_1 = (\lambda_1 - \lambda_2)/(\lambda_2 - \lambda_n)$, $c_{k-1}(x)$ 为 $k-1$ 次 Chebyshev 多项式.*

**证明** 由定理 8.1.2 知,

$$\theta_1 = \max_{y\neq 0} \frac{y^T T_k y}{y^T y} = \max_{y\neq 0} \frac{(Q_k y)^T A (Q_k y)}{(Q_k y)^T (Q_k y)}$$

$$= \max_{0\neq w\in K(A, q_1, k)} \frac{w^T A w}{w^T w}.$$

由于 $\lambda_1$ 是 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{w}/\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$ 对所有非零的 $\boldsymbol{w}$ 取最大, 故 $\lambda_1 \geqslant \theta_1$. 为了得到 $\theta_1$ 的下界, 考察下式:

$$\theta_1 = \max_{p\in P_{k-1}} \frac{\boldsymbol{q}_1^{\mathrm{T}} p(\boldsymbol{A})\boldsymbol{A}p(\boldsymbol{A})\boldsymbol{q}_1}{\boldsymbol{q}_1^{\mathrm{T}} p(\boldsymbol{A})^2 \boldsymbol{q}_1},$$

其中 $P_{k-1}$ 是所有不高于 $k-1$ 次的多项式组成的集合. 若 $q_1 = \sum_{i=1}^{n} d_i z_i$, 则

$$\begin{aligned}\frac{\boldsymbol{q}_1^{\mathrm{T}} p(\boldsymbol{A})\boldsymbol{A}p(\boldsymbol{A})\boldsymbol{q}_1}{\boldsymbol{q}_1^{\mathrm{T}} \boldsymbol{p}(\boldsymbol{A})^2 q_1} &= \frac{\sum_{i=1}^{n} d_i^2 p(\lambda_i)^2 \lambda_i}{\sum_{i=1}^{n} d_i^2 p(\lambda_i)^2} \\ &\geqslant \lambda_1 - (\lambda_1 - \lambda_n)\frac{\sum_{i=2}^{n} d_i^2 p(\lambda_i)^2}{d_1^2 p(\lambda_1)^2 + \sum_{i=2}^{n} d_i^2 p(\lambda_i)^2}.\end{aligned}$$

为了能得到一个较好的下界, 我们选取多项式 $p(x)$ 使得它在 $x = \lambda_1$ 的值比在其他特征值上的值大得多. 一种方式是选取

$$p(x) = c_{k-1}\left(-1 + 2\frac{x-\lambda_n}{\lambda_2 - \lambda_n}\right),$$

其中 $c_{k-1}(z)$ 是 $k-1$ 次 Chebyshev 多项式, 它由下列递推公式产生:

$$c_k(z) = 2zc_{k-1}(z) - c_{k-2}(z), \quad c_0 = 1, \quad c_1 = z.$$

这些多项式在区间 $[-1,1]$ 上的上界为 1, 但在区间外增长很快. 用这种方法定义的 $p(x)$ 有性质:

$$|p(\lambda_i)| \leqslant 1, \quad i = 2:n,$$

但 $p(\lambda_1) = c_{k-1}(1+2\rho_1)$. 故有

$$\theta_1 \geqslant \lambda_1 - (\lambda_1 - \lambda_n)\frac{1-d_1^2}{d_1^2}\cdot\frac{1}{c_{k-1}(1+2\rho_1)^2}.$$

注意到 $\tan^2(\phi_1) = (1-d_1^2)/d_1^2$, 上式即为所希望的下界. □

从此定理, 立即可得关于 $\theta_k$ 的一个类似的结果.

**推论 9.1.4** 采用上述定理的记号, 有下列结果:

$$\lambda_n \leqslant \theta_k \leqslant \lambda_n + \frac{(\lambda_1 - \lambda_n)\tan(\phi_n)^2}{c_{k-1}(1+2\rho_n)^2},$$

其中 $\rho_n = (\lambda_{n-1} - \lambda_n)/(\lambda_1 - \lambda_{n-1}), \cos(\phi_n) = q_n^{\mathrm{T}} z_n$.

**证明** 在定理 9.1.3 中, 用 $-\boldsymbol{A}$ 替换 $\boldsymbol{A}$, 即得. □

### 9.1.5 幂法和 Lanczos 方法的比较

值得将 $\boldsymbol{\theta}_1$ 与幂法的估计值 $\lambda_1$ 进行比较. (见 8.2.1 节.) 为简单起见, 假定 $\lambda_1 \geqslant \cdots \geqslant \lambda_n \geqslant 0$. 对 $q_1$ 用 $k-1$ 步幂法迭代后, 得到方向

$$\boldsymbol{v} = \boldsymbol{A}^{k-1}\boldsymbol{q}_1 = \sum_{i=1}^{n} c_i \lambda_i^{k-1} \boldsymbol{z}_i$$

上的一个向量, 并且有一个特征值估计

$$\gamma_1 = \frac{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{v}}{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}}.$$

利用定理 9.1.3 的证明和记号, 易得不等式:

$$\lambda_1 \geqslant \gamma_1 \geqslant \lambda_1 - (\lambda_1 - \lambda_n)\tan^2(\phi_1)\cdot\left(\frac{\lambda_2}{\lambda_1}\right)^{2k-1}. \tag{9.1.5}$$

(提示: 在证明中令 $p(x) = x^{k-1}$.) 因此, 我们可以比较 $\theta_1$ 和 $\gamma_1$ 的下界:

$$L_{k-1} \equiv 1/\left[c_{k-1}\left(2\frac{\lambda_1}{\lambda_2} - 1\right)\right]^2 \geqslant 1/[c_{k-1}(1+2\rho_1)]^2,$$

$$R_{k-1} = \left(\frac{\lambda_2}{\lambda_1}\right)^{2(k-1)}.$$

表 9.1.1 给出了两个下界对不同 $k$ 与 $\lambda_2/\lambda_1$ 的比较:

**表 9.1.1 $L_{k-1}/R_{k-1}$**

| $\lambda_1/\lambda_2$ | $k=5$ | $k=10$ | $k=15$ | $k=20$ | $k=25$ |
|---|---|---|---|---|---|
| 1.50 | $\frac{1.1\times10^{-4}}{3.9\times10^{-2}}$ | $\frac{2.0\times10^{-10}}{6.8\times10^{-4}}$ | $\frac{3.9\times10^{-16}}{1.2\times10^{-5}}$ | $\frac{7.14\times10^{-22}}{2.0\times10^{-7}}$ | $\frac{1.4\times10^{-27}}{3.5\times10^{-9}}$ |
| 1.10 | $\frac{2.7\times10^{-2}}{4.7\times10^{-1}}$ | $\frac{5.5\times10^{-5}}{1.8\times10^{-1}}$ | $\frac{1.1\times10^{-7}}{6.9\times10^{-2}}$ | $\frac{2.1\times10^{-10}}{2.7\times10^{-2}}$ | $\frac{4.2\times10^{-13}}{1.0\times10^{-2}}$ |
| 1.01 | $\frac{5.6\times10^{-1}}{9.2\times10^{-1}}$ | $\frac{1.0\times10^{-1}}{8.4\times10^{-1}}$ | $\frac{1.5\times10^{-2}}{7.6\times10^{-1}}$ | $\frac{2.0\times10^{-3}}{6.9\times10^{-1}}$ | $\frac{2.8\times10^{-4}}{6.2\times10^{-1}}$ |

Lanczos 估计的优越性是不言而喻的. 这并不奇怪, 因为 $\theta_1$ 是 $r(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}/\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}$ 对 $K(\boldsymbol{A}, \boldsymbol{q}_1, k)$ 中所有元素取最大值而得到的估计, 而 $\gamma_1 = r(\boldsymbol{v})$ 只是对应空间 $K(\boldsymbol{A}, \boldsymbol{q}_1, k)$ 中一特定的值, 即 $\boldsymbol{v} = \boldsymbol{A}^{k-1}\boldsymbol{q}_1$.

### 9.1.6 内部特征值的收敛性

在本节最后, 我们给出关于 $\boldsymbol{T}_k$ 内部特征值的误差界的一些注释. 定理 9.1.3 的证明的核心思想是利用变换了的 Chebyshev 多项式. 通过它, 我们放大了 $\boldsymbol{q}_1$ 在 $z_1$ 方向上的分量. 一个类似的想法可用来获得内部 Ritz 值 $\theta_i$ 的误差界. 然而, 这些误差界并不令人满意, 因为"放大多项多"具有形式 $q(x)\prod_{i=1}^{k-1}(x-\lambda_i)$, 其中 $q(x)$ 是在区间 $[\lambda_{i+1}, \lambda_n]$ 上的 $k-1$ 次 Chebyshev 多项式. 详细的讨论见 Kaniel(1966), Paige(1971) 或 Saad(1980).

## 习 题

**9.1.1** 设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 为反对称矩阵, 试推导一个计算反对称三对角矩阵 $\boldsymbol{T}_m$ 的 Lanczos 型算法, 使得 $\boldsymbol{A}\boldsymbol{Q}_m = \boldsymbol{Q}_m\boldsymbol{T}_m$, 其中 $\boldsymbol{Q}_m^{\mathrm{T}}\boldsymbol{Q}_m = \boldsymbol{I}_m$.

**9.1.2** 设 $A \in \mathbb{R}^{n\times n}$ 为对称矩阵, 定义函数 $r(x) = x^{\mathrm{T}}Ax/x^{\mathrm{T}}x$. $S \subseteq \mathbb{R}^n$ 是一子空间. 且任何 $x \in S$, 有 $\nabla r(x) \in S$, 证明 $S$ 是 $A$ 的不变子空间.

**9.1.3** 证明: 若对称矩阵 $A \in \mathbb{R}^{n\times n}$ 有一多重特征值. 则 Lanczos 迭代提前结束.

**9.1.4** 定理 9.1.1 中的指标 $m$ 是包含向量 $q_1$ 的 $A$ 的最小不变子空间.

**9.1.5** 已知 $A \in \mathbb{R}^{n\times n}$ 对称, 考虑如下问题: 确定正交序列 $q_1, q_2, \cdots, q_k, \cdots$ 使得一旦 $Q_k = [q_1, \cdots, q_k]$ 是已知的, $q_{k+1}$ 会使 $\mu_k = \|(I - Q_{k+1}Q_{k+1}^{\mathrm{T}})AQ_k\|_F$ 达到最小. 并证明: 若 $\mathrm{span}\{q_1, \cdots, q_k\} = K(A, q_1, k)$, 则可能选取 $q_{k+1}$ 使得 $\mu_k = 0$. 解释这一优化问题如何导出 Lanczos 迭代.

**9.1.6** 设 $A \in \mathbb{R}^{n\times n}$ 对称, 我们希望计算出它的最大特征值. 令 $\eta$ 为一近似特征向量, $\alpha = (\eta^{\mathrm{T}}A\eta)/(\eta^{\mathrm{T}}\eta), z = A\eta - \alpha\eta$. (a) 试证: 区间 $[\alpha-\delta, \alpha+\delta]$ 必包含 $A$ 的一个特征值, 其中 $\delta = \|z\|_2/\|\eta\|_2$. (b) 考虑新的逼近 $\bar{\eta} = a\eta + bz$, 如何确定实数 $a$, $b$, 使得 $\bar{\alpha} = (\bar{\eta}^{\mathrm{T}}A\bar{\eta})/(\bar{\eta}^{\mathrm{T}}\bar{\eta})$ 最大. (c) 阐述上述计算与 Lanczos 算法前二步的关系.

## 本节注释与参考文献

Lanczos 方法的经典参考文献如下.

C. Lanczos(1950). "An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators," J. *Res. Nat. Bur. Stand.* 45, 255–282.

尽管上文提到了 Ritz 值的收敛性，为更详细的了解，我们推荐下列参考文献.

S. Kaniel(1966). "Estimates for Some Computational Techniques in Linear Algebra," *Math. Comp.* 20, 369–378.

C. C. Paige(1971). "The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices," Ph. D. thesis, London University.

Y. Saad(1980). "On the Rates of Convergence of the Lanczos and the Block Lanczos Methods," *SIAM J. Num. Anal.* 17, 687–706.

有关 Lanczos 算法、正交多项式和矩理论之间联系的讨论可见下列参考文献.

N. J. Lehmann(1963). "Optimale Eigenwerteinschliessungen," *Numer. Math.* 5, 246–272.

A. S. Householder(1968). "Moments and characteristic Roots II," *Numer. Math.* 11, 126–128.

G. H. Golub(1974). "Some Uses of the Lanczos Algorithm in Numerical Linear Algebra," *in Topics in Numerical Analysis*, ed. , J. J. H. Miller, Academic Press, New York.

我们是通过讨论, 用 Householder 变换或 Givens 变换三对角化必然有非零元填充来研究 Lanczos 方法. 事实上，如果小心处理有时可将填充限制到可接受的水准.

I. S. Duff(1974). "Pivot Selection and Row Ordering in Givens Reduction on Sparse Matrices," *Computing* 13, 239–248.

I. S. Duff and J. K. Reid(1976). "A Comparison of Some Methods for the Solution of Sparse Over-Determined Systems of Linear Equations," *J. Inst. Maths. Applic.* 17, 267–280.

L. Kaufman(1979). "Application of Dense Householder Transformations to a Sparse Matrix," *ACM Trans. Math. Soft.* 5, 442–450.

## 9.2 实用 Lanczos 方法

舍入误差严重影响 Lanczos 迭代的表现. 其根本的困难在于 Lanczos 向量之间会失去正交性, 这个现象会搅乱算法的终止问题, 并且使得矩阵 $\boldsymbol{A}$ 的特征值和三对角矩阵 $\boldsymbol{T}_k$ 的特征值之间的关系复杂化. 这一点以及具有完美稳定性 Householder 三对角化方法的出现, 说明了为什么 Lanczos 方法在 20 世纪 50 年代与 60 年代被数值分析专家所忽视. 然而, 随着 Kaniel-Paige 理论的发展, 人们对此方法的兴趣又恢复了, 因为伴随计算机运算能力的不断提高, 解决大型稀疏矩阵的特征值问题迫在眉睫. 通常只用比 $n$ 少得多的迭代次数就能得到很好的两端特征值的近似值, 这使 Lanczos 方法作为稀疏矩阵技巧极富吸引力, 而不是作为 Householder 方法的"竞争者".

要成功地实现 Lanczos 迭代, 涉及的算法远不止迭代算法 (9.1.3) 那样简单. 在本节中, 我们概括地介绍几种被建议用来使 Lanczos 方法实际可行的实用思想.

### 9.2.1 精确的运算实现

仔细重写 (9.1.3) 且利用公式:

$$\alpha_k = \boldsymbol{q}_k^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{q}_k - \beta_{k-1}\boldsymbol{q}_{k-1}),$$

则整个 Lanczos 算法可用 2 个 $n$ 维向量的存储量来实现.

**算法 9.2.1(Lanczos 算法)** 给定一对称矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 2 范数单位向量 $w \in \mathbb{R}^n$, 本算法计算出一个 $k\times k$ 对称三对角矩阵 $\boldsymbol{T}_k$, 且 $\lambda(\boldsymbol{T}_k) \subset \lambda(\boldsymbol{A})$, 这里假定已有计算矩阵与向量乘积 $\boldsymbol{A}w$ 的函数**A.mult**$(w)$. $\boldsymbol{T}_k$ 的对角元和次对角元分别存储在 $\alpha(1:k)$ 与 $\beta(1:k-1)$.

$v(1:n) = 0; \beta_0 = 1; k = 0$
**while** $\beta_k \neq 0$
  **if** $k \neq 0$,
    **for** $i = 1:n$
      $t = w_i;\ w_i = v_i/\beta_k;\ v_i = -\beta_k t$
    **end**
  **end**
  $v = v + \mathbf{A.mult}(w)$
  $k = k + 1; \alpha_k = w^{\mathrm{T}}v; v = v - \alpha_k w; \beta_k = \|v\|_2$
**end**

注意, $\boldsymbol{A}$ 在整个过程中没有改变. 只需要提供一个计算矩阵 $\boldsymbol{A}$ 与向量乘积的程序**A.mult**$(\cdot)$. 如果矩阵 $\boldsymbol{A}$ 平均每行有 $i$ 个非零元, 则在每一 Lanczos 步中, 约有 $(2i+8)n$ 个 flop.

上述算法停止后, $T_k$ 的特征值可用对称三对角 QR 方法或 8.5 节中任何特殊的方法 (如二分法) 来计算.

Lanczos 向量通过 $n$ 维向量 $w$ 来产生. 如果这些向量以后要用到, 就必须对它们进行存储. 在典型的稀疏矩阵的情况, 它们可存储在磁盘或其他的辅助存储设备, 直到不需要为止.

### 9.2.2 误差性质

要开发一个实际的且易于使用的 Lanczos 程序, 需要利用 Paige(1971, 1976, 1980) 具有基石般作用的误差分析. 对他所得的结果进行观察, 是启发本节的几个修正 Lanczos 算法的最好途径.

算法进行 $j$ 步后, 我们得到由计算出的 Lanczos 向量组成的矩阵 $\hat{Q}_k=[\hat{q}_1,\hat{q}_2,\cdots,\hat{q}_k]$ 及相应的三对角矩阵:

$$\hat{T}_k=\begin{bmatrix} \hat{\alpha}_1 & \hat{\beta}_1 & & \cdots & 0 \\ \hat{\beta} & \hat{\alpha}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \hat{\beta}_{k-1} \\ 0 & \cdots & & \hat{\beta}_{k-1} & \hat{\alpha}_k \end{bmatrix}.$$

Paige(1971, 1976) 证明了, 如果 $\hat{r}_k$ 是 $r_k$ 的计算值, 则

$$A\hat{Q}_k=\hat{Q}_k\hat{T}_k+\hat{r}_k e_k^{\mathrm{T}}+E_k, \tag{9.2.1}$$

其中

$$\|E_k\|_2\approx \mathbf{u}\|A\|_2. \tag{9.2.2}$$

这说明在机器精度的意义下, 重要的方程 $AQ_k=Q_kT_k+r_ke_k^{\mathrm{T}}$ 能得到满足.

不幸的是, $\hat{q}_i$ 之间的正交性远不如上面的结果那样美好. (正规性不成问题, 因为 Lanczos 向量的计算值本质上具有单位长度). 若 $\hat{\beta}_k=fl(\|\hat{r}_k\|_2)$, 计算 $\hat{q}_{k+1}=fl(\hat{r}_k/\hat{\beta}_k)$, 则简单的分析可证明 $\hat{\beta}_k\hat{q}_{k+1}\approx\hat{r}_k+w_k$, 其中 $\|w_k\|_2\approx\mathbf{u}\|\hat{r}_k\|_2\approx\mathbf{u}\|A\|_2$. 因此, 我们可得

$$|\hat{q}_{k+1}^{\mathrm{T}}\hat{q}_i|\approx\frac{|\hat{r}_k^{\mathrm{T}}\hat{q}_i|+\mathbf{u}\|A\|_2}{|\hat{\beta}_k|},\quad i=1:k.$$

换句话说, 当 $\beta_k$ 很小时, 可能会出现与正交性很大的偏离, 既使在 $\hat{r}_k^{\mathrm{T}}\hat{Q}_k=0$ 这一理想情况下也是如此. $\hat{\beta}_k$ 很小意味着 $\hat{r}_k$ 的计算将有“相消”. 我们要强调的是, 正交性的丢失是由于这种“相消”所引起的, 而不是舍入误差积累的结果.

**例 9.2.1** 矩阵

$$A=\begin{bmatrix} 2.64 & -0.48 \\ -0.48 & 2.36 \end{bmatrix}$$

的特征值为 $\lambda_1 = 3, \lambda_2 = 2$. 如果对 $\boldsymbol{A}$ 用 Lanczos 算法, 取初始向量为 $\boldsymbol{q}_1 = [0.810, -0.586]^{\mathrm{T}}$, 用三位浮点运算, 则得 $\hat{q}_2 = [0.707, 0.707]^{\mathrm{T}}$. 正交性失去了, 因为 $\mathrm{span}\{q_1\}$ 几乎就是 $\boldsymbol{A}$ 的不变子空间. (向量 $\boldsymbol{x} = [0.8, -0.6]^{\mathrm{T}}$ 是对应于 $\lambda_1$ 的特征向量.)

稍后我们将给出 Paige 分析的更多细节. 现在只需说在实际中总是失去正交性, 从而明显破坏 $\hat{\boldsymbol{T}}_k$ 的特征值的准确性. 这可通过 (9.2.1) 和定理 8.1.16 来量化. 特别地, 在定理中取 $\boldsymbol{F}_1 = \hat{r}_k \boldsymbol{e}_k^{\mathrm{T}} + \boldsymbol{E}_k, \boldsymbol{X}_1 = \hat{\boldsymbol{Q}}_k, \boldsymbol{S} = \hat{\boldsymbol{T}}_k$, 并且假定

$$\tau = \|\hat{\boldsymbol{Q}}_k^{\mathrm{T}} \hat{\boldsymbol{Q}}_k - \boldsymbol{I}_k\|_2$$

满足 $\tau < 1$. 则存在 $\mu_1, \cdots, \mu_k \in \lambda(\boldsymbol{A})$, 使得

$$|\mu_i - \lambda_i(\boldsymbol{T}_k)| \leqslant \sqrt{2}(\|\hat{\boldsymbol{r}}_k\|_2 + \|\boldsymbol{E}_k\|_2 + \tau(2+\tau)\|\boldsymbol{A}\|_2), \quad i = 1:k.$$

控制因子 $\tau$ 的一个明显的办法是把新计算的 Lanczos 向量与以前所计算的 Lanczos 向量正交化. 这导致了第一个“实用”的 Lanczos 算法.

### 9.2.3 完全再正交化的 Lanczos 方法

给定 $\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1} \in \mathbb{R}^n$, 并假定已计算出 Householder 矩阵 $\boldsymbol{H}_0, \boldsymbol{H}_1, \cdots, \boldsymbol{H}_{k-1}$ 使得 $(\boldsymbol{H}_0\boldsymbol{H}_1\cdots\boldsymbol{H}_{k-1})^{\mathrm{T}}[\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_{k-1}]$ 为上三角形矩阵. 用 $[\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k]$ 表示 Householder 矩阵乘积 $(\boldsymbol{H}_0\boldsymbol{H}_1\cdots\boldsymbol{H}_{k-1})$ 的前 $k$ 列. 现假定给出向量 $\boldsymbol{r}_k \in \mathbb{R}^n$, 我们希望在下述方向上得到一单位向量 $q_{k+1}$:

$$w = \boldsymbol{r}_k - \sum_{i=1}^{k} (\boldsymbol{q}_i^{\mathrm{T}} \boldsymbol{r}_k)\boldsymbol{q}_i \in \mathrm{span}\{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k\}^{\perp}.$$

如果选取 Householder 矩阵 $\boldsymbol{H}_k$ 使得 $(\boldsymbol{H}_0\boldsymbol{H}_1\cdots\boldsymbol{H}_k)^{\mathrm{T}}[\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k]$ 为上三角形矩阵, 则 $\boldsymbol{H}_0\boldsymbol{H}_1\cdots\boldsymbol{H}_k$ 的第 $k+1$ 列就是我们所需要单位向量.

如果把这些 Householder 计算和 Lanczos 算法结合起来, 就可得到与机器精度无关的 Lanczos 向量.

$r_0 = q_1$(给定的单位向量)
确定 Householder 矩阵 $H_0$ 使得 $H_0 r_0 = e_1$
$\alpha_1 = q_1^{\mathrm{T}} A q_1$
**for** $k = 1 : n-1$ (9.2.3)
  $r_k = (A - \alpha_k I)q_k - \beta_{k-1} q_{k-1} (\beta_0 q_0 \equiv 0)$
  $\omega = (H_{k-1} \cdots H_0) r_k$
  确定 Householder 矩阵 $H_k$ 使得 $H_k w = (w_1, \cdots, w_k, \beta_k, 0, \cdots, 0)^{\mathrm{T}}$
  $q_{k+1} = H_0 H_1 \cdots H_k e_{k+1}$; $\alpha_{k+1} = q_{k+1}^{\mathrm{T}} A q_{k+1}$
**end**

这是一个完全再正交化 Lanczos 算法的例子. 更加深入的分析见 Paige(1970). 用

Householder 矩阵来加强正交性的思想见 Golub, Underwood, and Wilkinson(1972).

由 (9.2.3) 计算出的 $\hat{q}_i$ 与机器精度无关, 这一点可从 Householder 矩阵的误差性质中看出, 注意到, 由 $q_{k+1}$ 的定义, $\beta_k = 0$ 是没有影响的. 由于这一原因, 算法可安全地运行到 $k = n - 1$ 步 (然而, 在实际中, 算法在一个小得多的 $k$ 值就停止).

当然, 无论 (9.2.3) 如何实现, 只存储 Householder 向量 $v_k$, 而不会显式地形成相应的 $P_k$. 既然 $\boldsymbol{H}_k(1:k, 1:k)=I_k$, 我们没有必要去计算 $\boldsymbol{\omega} = (\boldsymbol{H}_{k-1}, \boldsymbol{H}_{k-2} \cdots \boldsymbol{H}_0) r_k$ 的前 $k$ 个分量, 因为精确的运算会使这些分量为零.

不幸的是, 在完全再正交化的计算中, 这些措施的意义并不大. 因为在 Lanczos 算法的第 $k$ 步, 计算 Householder 矩阵会增加 $O(kn)$ 个 flop. 此外, 为了计算 $q_{k+1}$, 也要用到相应于 $\boldsymbol{H}_0, \boldsymbol{H}_1, \cdots, \boldsymbol{H}_k$ 的 Householder 向量. 当 $n$ 和 $k$ 很大时, 这通常意味着无法接受的数据传输量.

因此, 完全再正交化要付出很高的代价. 所幸的是, 有更加有效的算法可取, 但这要求我们更加深入了解正交性是怎样失去的.

### 9.2.4 有选择的正交化

Paige(1971) 误差分析的一个惊人的且使人啼笑皆非的结论是: 正交性的丢失与 Ritz 对的收敛性是密不可分的. 确切地说, 假定对 $\hat{\boldsymbol{T}}_k$ 用 QR 算法, 得到 Ritz 值的计算解 $\hat{\theta}_1, \cdots, \hat{\theta}_k$ 和一个由特征向量组成的几乎正交的矩阵 $\hat{\boldsymbol{S}}_k = (\hat{s}_{pq})$. 若 $\hat{\boldsymbol{Y}}_k = [\hat{\boldsymbol{y}}_1, \cdots, \hat{\boldsymbol{y}}_k] = fl(\hat{\boldsymbol{Q}}\hat{\boldsymbol{S}}_k)$, 则可证明,

$$|\hat{\boldsymbol{q}}_{k+1}^{\mathrm{T}} \hat{\boldsymbol{y}}_i| \approx \frac{\mathbf{u}\|\boldsymbol{A}\|_2}{|\hat{\beta}_k||\hat{s}_{ki}|}, \quad i = 1:k, \tag{9.2.4}$$

$$\|\boldsymbol{A}\hat{\boldsymbol{y}}_i - \hat{\theta}_i \hat{\boldsymbol{y}}_i\|_2 \approx |\hat{\beta}_k||\hat{s}_{ki}|. \tag{9.2.5}$$

也就是说, 最新计算出的 Lanczos 向量 $\hat{q}_{k+1}$ 倾向于在任何已收敛的 Ritz 向量的方向上含有不可被忽视但不需要的非零分量. 因此, 我们不去将 $\hat{\boldsymbol{q}}_{k+1}$ 与以前所有计算出的 Lanczos 向量正交化, 而是通过让它与一个小得多的由收敛的 Ritz 向量组成的集合正交, 以达到同样的效果.

Parlett and Scott(1979) 讨论了用这种途径来加强正交性的实现问题. 在他们的称为*有选择的正交化*技巧中, 一个计算的 Ritz 对 $(\hat{\theta}, \boldsymbol{y})$ 被称为是"好的", 如果它满足:

$$\|\boldsymbol{A}\hat{\boldsymbol{y}} - \hat{\theta}\hat{\boldsymbol{y}}\|_2 \approx \sqrt{\mathbf{u}}\|\boldsymbol{A}\|_2.$$

一旦 $\hat{\boldsymbol{q}}_{k+1}$ 已被算出, 把它对每个"好的"Ritz 向量正交化. 这比完全重新正交化效率高得多, 因为通常情况下好的 Ritz 向量比 Lanczos 向量少很多.

一种实现选择正交化的途径是在每一步对角化 $\hat{\boldsymbol{T}}_k$, 并根据 (9.2.4) 与 (9.2.5) 来观察 $\hat{s}_{ki}$. 一种更有效的方式是用以下结果来估计正交性丢失的度量 $\|\boldsymbol{I}_k - \hat{\boldsymbol{Q}}_k^{\mathrm{T}} \hat{\boldsymbol{Q}}_k\|_2$.

**引理 9.2.1**　假定 $S_+ = [S, d], S \in \mathbb{R}^{n\times k}, d \in \mathbb{R}^n$, 如果 $S$ 满足 $\|I_k - S^{\mathrm{T}}S\|_2 \leqslant \mu$, $|1 - d^{\mathrm{T}}d| \leqslant \delta$, 则 $\|I_{k+1} - S_+^{\mathrm{T}}S_+\|_2 \leqslant \mu_+$, 其中

$$\mu_+ = \frac{1}{2}(\mu + \delta + \sqrt{(\mu-\delta)^2 + 4\|S^{\mathrm{T}}d\|_2^2}).$$

**证明**　参见 Kahan and Parlett(1974) 或 Parlett and Scott(1979).　□

因此, 当我们已知 $\|I_k - \hat{Q}_k^{\mathrm{T}}\hat{Q}_k\|_2$ 的界时, 就可以对 $S = \hat{Q}_k, d = \hat{q}_{k+1}$ 用以上引理得出 $\|I_{k+1} - \hat{Q}_{k+1}^{\mathrm{T}}\hat{Q}_{k+1}\|_2$ 的界. (在这种情形, $\delta \approx \mathbf{u}$, 且假定 $\hat{q}_{k+1}$ 已对当前好的 Ritz 向量集正交化了). 不需要利用 $\hat{q}_1, \cdots, \hat{q}_k$, 而只通过一简单迭代就可估计出 $\hat{Q}_k^{\mathrm{T}}q_{k+1}$ 的范数, 见 Kahan and Parlett(1974) 或 Parlett and Scott(1979), 额外的开销是很小的, 而且当估计界显示已失去正交性时, 就要考虑扩大好的 Ritz 向量的集合. 之后, 并且只有在这以后, 将 $\hat{T}_k$ 对角化.

### 9.2.5　幽灵特征值问题

在设法构造一个不涉及任何强迫正交性的可行 Lanczos 算法方面, 已付出了巨大的努力. 这方面研究集中在"幽灵"或"虚假"特征值问题. 这是指 $\hat{T}_k$ 的多重特征值, 它们对应于 $A$ 的单重特征值. 当失去与收敛的 Ritz 向量的正交性后, 迭代实际上重新开始, 所以会出现这些多重特征值. (打个比方, 在 8.2.8 节的正交迭代中, 如果忘记了正交化, 想象一下会发生什么情况.)

Collum and Willough(1979) 以及 Parlett and Reid(1981) 讨论了鉴别和处理这些幽灵特征值的问题. 在需要 $A$ 的全部特征值的应用中, 这是十分紧迫的问题, 因为上述正交化过程代价高得无法实现.

既使 $A$ 有一个真正的多重特征值, Lanczos 迭代出现困难也是意料之中的, 这是由于 $\hat{T}_k$ 是不可约的, 而不可约的三对角矩阵不可能有多重特征值. 我们下一个实用的 Lanczos 算法就是要绕过这一困难.

### 9.2.6　分块 Lanczos 方法

就像简单的幂法有同时迭代的块形式一样, Lanczos 算法也有块形式. 假定 $n = rp$, 考虑以下分解:

$$Q^{\mathrm{T}}AQ = \bar{T} = \begin{bmatrix} M_1 & B_1^{\mathrm{T}} & & \cdots & 0 \\ B_1 & M_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & B_{r-1}^{\mathrm{T}} \\ 0 & \cdots & & B_{r-1} & M_r \end{bmatrix}, \tag{9.2.6}$$

其中 $Q = [X_1, \cdots, X_r], X_i \in \mathbb{R}^{n\times p}$ 互相正交, $M_i \in \mathbb{R}^{p\times p}$, $B_i \in \mathbb{R}^{p\times p}$ 为上三角形矩阵. 比较 $AQ = Q\bar{T}$ 两边的每一块可得, 对于 $k = 1 : r-1$,

$$\boldsymbol{A}\boldsymbol{X}_k = \boldsymbol{X}_{k-1}\boldsymbol{B}_{k-1}^{\mathrm{T}} + \boldsymbol{X}_k\boldsymbol{M}_k + \boldsymbol{X}_{k+1}\boldsymbol{B}_k, \quad \boldsymbol{X}_0\boldsymbol{B}_0 \equiv 0.$$

从 $\boldsymbol{Q}$ 的正交性, 我们有

$$\boldsymbol{M}_k = \boldsymbol{X}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}_k, \quad k = 1 : r.$$

此外, 若令 $\boldsymbol{R}_k = \boldsymbol{A}\boldsymbol{X}_k - \boldsymbol{X}_k\boldsymbol{M}_k - \boldsymbol{X}_{k-1}\boldsymbol{B}_{k-1}^{\mathrm{T}} \in \mathbb{R}^{n\times p}$, 则 $\boldsymbol{X}_{k+1}\boldsymbol{B}_k = \boldsymbol{R}_k$ 是 $\boldsymbol{R}_k$ 的 QR 分解. 这些关系式建议我们用如下方法产生 (9.2.6) 中的块三对角矩阵:

$$\begin{aligned}
&\text{给定 } X_1 \in \mathbb{R}^{p\times p}, X_1^{\mathrm{T}}X_1 = I_p \\
&M_1 = X_1^{\mathrm{T}}AX_1 \\
&\textbf{for } k = 1 : r-1 \\
&\qquad R_k = AX_k - X_kM_k - X_{k-1}B_{k-1}^{\mathrm{T}} \quad (X_0B_0^{\mathrm{T}} \equiv 0) \\
&\qquad X_{k+1}B_k = R_k \ (\text{QR 分解}) \\
&\qquad M_{k+1} = X_{k+1}^{\mathrm{T}}AX_{k+1} \\
&\textbf{end}
\end{aligned} \tag{9.2.7}$$

在第 $k$ 次循环的开始, 我们有

$$\boldsymbol{A}[\boldsymbol{X}_1, \cdots, \boldsymbol{X}_k] = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_k]\bar{\boldsymbol{T}}_k + \boldsymbol{R}_k[0, \cdots, 0, \boldsymbol{I}_p], \tag{9.2.8}$$

其中

$$\bar{\boldsymbol{T}}_k = \begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{B}_1^{\mathrm{T}} & & \cdots & \boldsymbol{0} \\ \boldsymbol{B}_1 & \boldsymbol{M}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{B}_{k-1}^{\mathrm{T}} \\ \boldsymbol{0} & \cdots & & \boldsymbol{B}_{k-1} & \boldsymbol{M}_k \end{bmatrix}$$

利用与定理 9.1.1 的证明类似的方式, 我们可以证明, 只要所有的 $\boldsymbol{R}_k$ 都不是秩亏的, 则 $\boldsymbol{X}_k$ 是相互正交的. 然而, 如果对某个 $k$ 有 $\mathrm{rank}(\boldsymbol{R}_k) < p$, 则可选择 $\boldsymbol{X}_{k+1}$ 的列, 使得 $\boldsymbol{X}_{k+1}^{\mathrm{T}}\boldsymbol{X}_i = 0, i = 1 : k$, 见 Golub and Underwood(1997).

因为 $\bar{\boldsymbol{T}}_k$ 的带宽为 $p$, 用 Schwartz(1968) 的算法可有效地把它化成三对角矩阵. 一旦有了三对角形式, 用对称 QR 算法即可得到 Ritz 值.

为了明智地决定何时用块 Lanczos 方法, 有必要知道块的维数怎样影响 Ritz 值的收敛. 定理 9.1.3 的如下推广阐明了这一问题.

**定理 9.2.2** 设 $n\times n$ 对称矩阵 $\boldsymbol{A}$ 的特征值为 $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n$, 相应的规范正交特征向量为 $\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_n$. 假定 $\bar{\boldsymbol{T}}_k$ 为块 Lanczos 迭代 (9.2.7) 进行 $k$ 步后得到的块三对角矩阵, 它的 $p$ 个最大的特征值为 $\mu_1, \cdots, \mu_p$. 若令 $\boldsymbol{Z}_1 = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_p]$, 且 $\cos(\theta_p) = \sigma_p(\boldsymbol{Z}_1^{\mathrm{T}}\boldsymbol{X}_1) > 0$, 则

$$\lambda_i \geqslant \mu_i \geqslant \lambda_i - \varepsilon_i^2, \quad i = 1 : p,$$

其中

$$\varepsilon_i^2 = \frac{(\lambda_1 - \lambda_i)\tan^2(\theta_p)}{\left[c_{k-1}\left(\dfrac{1+\gamma_i}{1-\gamma_i}\right)\right]^2}, \quad \gamma_i = \frac{\lambda_i - \lambda_{p+1}}{\lambda_i - \lambda_n},$$

$c_{k-1}(z)$ 为 $k-1$ 次 Chebyshev 多项式.

**证明** 见 Underwood(1975) □

把上述定理中的 $\boldsymbol{A}$ 换成 $-\boldsymbol{A}$, 对 $\bar{\boldsymbol{T}}_k$ 的最小的特征值可得到类似的结果.

根据定理 9.2.2 和块 Lanczos 迭代 (9.2.7), 我们可总结出下面几条性质:

- 随着 $p$ 的增加, Ritz 值的误差界得到改善.
- 计算 $\bar{T}_k$ 的特征值的工作量与 $p^2$ 成正比.
- 块的维数应至少与任一需要计算的特征值之最大重数一样大.

Scott(1979) 详细讨论了在这些因素下如何决定块的维数.

正交性的丢失同样困扰块 Lanczos 算法. 然而, 前面介绍的强迫正交性的所有技巧都可推广到分块的情形.

### 9.2.7 $s$ 步 Lanczos 方法

可以按迭代的方式应用块 Lanczos 算法 (9.2.7) 来计算 $\boldsymbol{A}$ 的选定特征值. 为了阐明思想, 假定我们要计算 $p$ 个最大的特征值. 如果给定矩阵 $\boldsymbol{X}_1 \in \mathbb{R}^{n\times p}$ 是列正交的, 我们可用下面的方式计算.

**until** $\|AX_1 - X_1\bar{T}_S\|_F$ 足够小

通过块 Lanczos 算法, 产生 $X_2, \cdots, X_s \in \mathbb{R}^{n\times p}$;

形成 $sp \times sp$ 的 $p$ 对角矩阵

$$\bar{\boldsymbol{T}}_s = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_s]^{\mathrm{T}}\boldsymbol{A}[\boldsymbol{X}_1, \cdots, \boldsymbol{X}_s];$$

计算正交矩阵 $\boldsymbol{U} = [u_1, \cdots, u_{sp}]$ 使得

$$\boldsymbol{U}^{\mathrm{T}}\bar{\boldsymbol{T}}_s\boldsymbol{U} = \mathrm{diag}(\theta_1, \cdots, \theta_{sp}), \quad \theta_1 \geqslant \cdots \geqslant \theta_{sp}.$$

令 $\boldsymbol{X}_1 = [\boldsymbol{X}_1, \cdots, \boldsymbol{X}_s][u_1, \cdots, u_p]$

**end**

这就是分块的 $s$步Lanczos算法. Cullum and Donath(1974) 和 Underwood(1975) 对它进行了广泛的分析.

同样的思想也可以用来计算 $\boldsymbol{A}$ 的几个最小的特征值或最大最小特征值的混合情形, 见 Cullum(1978). 参数 $s$ 和 $p$ 的选取依赖于存储限制以及前面讨论块维数时提到的几个因素, 随着好的 Ritz 向量的出现, 块的维数 $p$ 可能减小. 然而这要求强迫与已收敛向量的正交, 见 Cullum and Donath(1974).

## 习 题

**9.2.1** 证明引理 9.2.1.

**9.2.2** 在 (9.2.7) 中, 若 $\text{rank}(\boldsymbol{R}_k) < p$, 象空间 $\text{range}([\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_k])$ 含有 $\boldsymbol{A}$ 的特征向量吗?

## 本节注释与参考文献

在 Lanczos 方法几种计算的变化形式中, 算法 9.2.1 最稳定, 详见:

C. C. Paige(1972) ."Computational Variants of the Lanczos Method for the Eigenproblem," *J. Inst. Math. Applic.* 10, 373–381.

关于 Lanczos 过程实现的其他实际细节的讨论见:

D. S. Scott(1979). "How to Make the Lanczos Algorithm Converge Slowly," *Math. Comp.* 33. 239–247.

B. N. Parlett, H. Simon,and L. M. Stringer(1982). "On Estimating the Largest Eigenvalue with the Lanczos Algorithm," *Math. Comp.* 38, 153–166.

B. N. Parlett and B. Nour-Omid(1985). "The Use of a Refined Error Bound When Updating Eigenvalues of Tridiagonals," *Lin. Alg. and Its Applic.* 68, 179–220.

J. Kuczyński and H. Woźniakowski(1992). "Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start," *SIAM J. Matrix Anal. Appl.* 13, 1094–1122.

在有舍入误差情况下 Lanczos 算法的表现最初出现在:

C. C. Paige(1971). "The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices," Ph. D. thesis, University of London.

随后的重要文章包括:

C. C. Paige(1976). "Error Analysis of the Lanczos Algorithm for Tridiagonalizing Symmetric Matrix," *J. Inst. Math. Applic.* 18, 341–349.

C. C. Paige(1980). "Accuracy and Effectiveness of the Lanczos Algorithm for the Symmetric Eigenproblem," *Lin. Alg. and Its Applic.* 34, 235–258.

关于不同的再正交化技巧可见:

C. C. Paige(1970). "Practical Use of the Symmetric Lanczos Process with Reorthogonalization," *BIT* 10, 183–195.

G. H. Golub, R. Underwood, and J. H. Wilkinson(1972). "The Lanczos Algorithm for the Symmetric $Ax = \lambda Bx$ Problem," Report STAN-CS-72-270, Department of Computer Science, Stanford University, Stanford, California.

B. N. Parlett and D. S. Scott(1979). "The Lanczos Algorithm with Selective Orthogonalization," *Math. Comp.* 33, 217–238.

H. Simon(1984). "Analysis of the Symmetric Lanczos Algorithm with Reorthogonalization Methods," *Lin. Alg. and Its Applic.* 61, 101–132.

不进行再正交化, 则有必要监测正交性的损失并在恰当的时候退出, 或者能设计某种技巧, 使之有助于区别幽灵特征值和实际特征值. 见:

W. Kahan and B. N. Parlett(1976). "How Far Should You Go with the Lanczos Process?" in

*Sparse Matrix Computations*, ed J. Bunch and D. Rose, Academic Press, New York, pp. 131–144.

J. Cullum and R. A. Willoughby(1979). "Lanczos and the Computation in Specified Intervals of the Spectrum of Large, Sparse Real Symmetric Matrices," in *Sparse Matrix Proc*, 1978, ed. I. S. Duff and G. W. Stewart, SIAM Publications, Philadelphia, PA.

B. N. Parlett and J. K. Reid(1981). "Tracking the Progress of the Lanczos Algorithm for Large Symmetric Eigenproblems," *IMA J. Num. Anal.* 1, 135–155.

D. Calvetti, L. Reichel, and D. C. Sorensen(1994). "An Implicitly Restarted Lanczos Method for Large Symmetric Eigenvalue Problems," *ETNA* 2, 1–21.

分块 Lanczos 算法的文献可见:

J. Cullum and W. E. Donath(1974). "A Block Lanczos Algorithm for Computing the $q$ Algebraically Largest Eigenvalues and a Corresponding Eigenspace of Large Sparse Real Symmetric Matrices," *Proc. of the* 1974 *IEEE Conf. on Decision and Control*, Phoenix, Arizona, pp. 505–509.

R. Underwood(1975). "An Iterative Block Lanczos Method for the Solution of Large Sparse Symmetric Eigenproblems," Report SIAN-CS-75-495, Department of Computer Science, Stanford University, Stanford, California.

G. H. Golub and R. Underwood(1977). "The Block Lanczos Method for Computing Eigenvalues," in *Mathematical Software* III, ed. J. Rice, Academic Press, New York, pp. 364–377.

J. Cullum(1978). "The Simultaneous Computation of a Few of the Algebraically Largest and Smallest Eigenvalues of a Large Sparse Symmetric Matrix," *BIT* 18, 265–275.

A. Ruhe(1979). "Implementation Aspects of Band Lanczos Algorithms for Computation of Eigenvalues of Large Sparse Symmetric Matrices," *Math. Comp.* 33, 680–687.

分块 Lanczos 算法产生一对称的带状矩阵, 它的特征值可用好几种方法来计算. 其中一种方法可见:

H. R. Schwartz(1968). "Tridiagonalization of a Symmetric Band Matrix," *Numer. Math.* 12, 231–241. See also Wilkinson and Reinsch(1971, 273–283).

在一些应用中, 有必要得出内部特征值的估计. 然而, Lanczos 算法倾向于先找出两端的特征值. 下面的文章讨论了这一问题:

A. K. Cline, G. H. Golub, and G. W. Platzman(1976). "Calculation of Normal Modes of Oceans Using a Lanczos Method," in *Sparse Matrix Computations*, ed, J. R. Bunch and D. J. Rose, Academic Press, New York, pp. 409–426.

T. Ericsson and A. Ruhe(1980). "The Spectral Transformation Lanczos Method for the Numerical Solution of Large Sparse Generalized Symmetric Eigenvalue Problems," *Math. Comp.* 35, 1251–1268.

R. G. Grimes, J. G. Lewis, and H. D. Simon(1994). "A Shifted Block Lanczos Algorthm for Solving Sparse Symmetric Generalized Eigenproblems." *SIAM J. Matrix Anal. Appl.* 15, 228–272.

# 9.3 应用于 $Ax=b$ 和最小二乘

在本节, 我们将简要地介绍如何把 Lanczos 算法用来解大型稀疏线性方程组与最小二乘问题. 更详细的内容, 见 Saunders(1995).

### 9.3.1 对称正定方程组

假设矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 对称正定, 考虑函数

$$\phi(x)=\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}-\boldsymbol{x}^{\mathrm{T}}\boldsymbol{b},$$

其中 $\boldsymbol{b}\in\mathbb{R}^n$. 因为 $\nabla\phi(\boldsymbol{x})=\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}$, 故 $\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$ 是 $\phi$ 的唯一极小点. 所以, $\phi$ 的一个近似极小点可看成是 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 的近似解.

给定初值 $\boldsymbol{x}_0\in\mathbb{R}^n$, 一种产生收敛于 $\boldsymbol{x}$ 的向量序列 $\{\boldsymbol{x}_k\}$ 的方法是对 $k=1:n$, 构造一列正交向量 $\{\boldsymbol{q}_k\}$, 且 $\boldsymbol{x}_k$ 为 $\phi$ 在集合

$$\boldsymbol{x}_0+\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}=\{\boldsymbol{x}_0+a_1\boldsymbol{q}_1+\cdots+a_k\boldsymbol{q}_k:a_k\in\mathbb{R}\}$$

中的极小点. 如果 $\boldsymbol{Q}_k=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k]$, 则这就意味着寻找向量 $\boldsymbol{y}\in\mathbb{R}^k$, 使得

$$\begin{aligned}\phi(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y})&=\frac{1}{2}(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y})^{\mathrm{T}}\boldsymbol{A}(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y})-(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y})^{\mathrm{T}}\boldsymbol{b}\\&=\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)\boldsymbol{y}-\boldsymbol{y}^{\mathrm{T}}\boldsymbol{Q}_k^{\mathrm{T}}(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0)+\phi(\boldsymbol{x}_0)\end{aligned}$$

达到极小. 观察该表达式对 $\boldsymbol{y}$ 的梯度, 可知

$$\boldsymbol{x}_k=\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y}_k,\tag{9.3.1}$$

其中 $\boldsymbol{y}_k$ 满足

$$(\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k)\boldsymbol{y}_k=\boldsymbol{Q}_k^{\mathrm{T}}(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0).\tag{9.3.2}$$

当 $k=n$ 时, $\phi$ 在整个空间 $\mathbb{R}^n$ 上达到极小, 故 $\boldsymbol{A}\boldsymbol{x}_n=\boldsymbol{b}$.

当 $\boldsymbol{A}$ 为大型稀疏矩阵时, 为了使求解过程有效, 需克服两大困难:

- 线性方程组 (9.3.2) 必须容易求解;
- 我们必须能够在计算 $\boldsymbol{x}_k$ 时不需要像 (9.3.1) 那样显式用到 $\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k$. 否则的话, 将会产生过量的数据流动.

我们将证明, 当 $\boldsymbol{q}_k$ 为 Lanczos 向量时, 这两大困难都将被克服.

Lanczos 算法进行 $k$ 步后, 我们得到分解:

$$\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{Q}_k\boldsymbol{T}_k+r_k\boldsymbol{e}_k^{\mathrm{T}},\tag{9.3.3}$$

其中

$$T_k = Q_k^{\mathrm{T}} A Q_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{k-1} \\ 0 & \cdots & & \beta_{k-1} & \alpha_k \end{bmatrix}. \tag{9.3.4}$$

通过这种途径, (9.3.2) 变成了一对称正定的三对角方程组, 它可以用 $\mathrm{LDL^T}$ 分解来迅速求解 (见算法 4.3.6). 特别地, 令

$$L_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \mu_1 & 1 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \mu_{k-1} & 1 \end{bmatrix}, \quad D_k = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & d_k \end{bmatrix}.$$

比较方程

$$T_k = L_k D_k L_k^{\mathrm{T}} \tag{9.3.5}$$

两边的元素, 可得下述算法:

$d_1 = \alpha_1$
**for** $i = 2 : k$
  $\mu_{i-1} = \beta_{i-1}/d_{i-1}$
  $d_i = \alpha_i - \beta_{i-1}\mu_{i-1}$
**end**

注意, 为了从 $L_{k-1}$ 和 $D_{k-1}$ 得到 $L_k$ 和 $D_k$, 我们只需计算数值

$$\mu_{k-1} = \beta_{k-1}/d_{k-1}, \quad d_k = \alpha_k - \beta_{k-1}\mu_{k-1}. \tag{9.3.6}$$

正如我们前面所提到的, 有效地计算 (9.3.1) 中的 $x_k$ 是非常关键的. 为此, 定义矩阵 $C_k \in \mathbb{R}^{n\times k}$ 和向量 $p_k \in \mathbb{R}^k$ 如下:

$$C_k L_k^{\mathrm{T}} = Q_k, \quad L_k D_k p_k = Q_k^{\mathrm{T}}(b - Ax_0). \tag{9.3.7}$$

我们发现, 若 $r_0 = b - Ax_0$, 则

$$x_k = x_0 + Q_k T_k^{-1} Q_k^{\mathrm{T}} r_0 = x_0 + Q_k (L_k D_k L_k^{\mathrm{T}})^{-1} Q_k^{\mathrm{T}} r_0 = x_0 + C_k p_k.$$

对 $C_k$ 作列划分 $C_k = [c_1, c_2, \cdots, c_k]$. 从 (9.3.7) 可得

$$[c_1, \mu_1 c_1 + c_2, \cdots, \mu_{k-1} c_{k-1} + c_k] = [q_1, \cdots, q_k].$$

因此, $C_k = [C_{k-1}, c_k]$, 其中

$$c_k = q_k - \mu_{k-1} c_{k-1}.$$

同样可观察到, 若在方程组 $L_k D_k p_k = Q_k^{\mathrm{T}} r_0$ 中令 $p_k = [\rho_1, \rho_2, \cdots, \rho_k]^{\mathrm{T}}$, 则方程组变为

$$\left[\begin{array}{c|c} \boldsymbol{L}_{k-1}\boldsymbol{D}_{k-1} & 0 \\ \hline 0 \ \cdots \ 0 \ \mu_{k-1}d_{k-1} & d_k \end{array}\right]\left[\begin{array}{c} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_{k-1} \\ \hline \rho_k \end{array}\right] = \left[\begin{array}{c} \boldsymbol{q}_1^{\mathrm{T}}\boldsymbol{r}_0 \\ \boldsymbol{q}_2^{\mathrm{T}}\boldsymbol{r}_0 \\ \vdots \\ \boldsymbol{q}_{k-1}^{\mathrm{T}}\boldsymbol{r}_0 \\ \hline \boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{r}_0 \end{array}\right].$$

既然 $\boldsymbol{L}_{k-1}\boldsymbol{D}_{k-1}\boldsymbol{p}_{k-1} = \boldsymbol{Q}_{k-1}^{\mathrm{T}}\boldsymbol{r}_0$, 可见

$$\boldsymbol{p}_k = \left[\begin{array}{c} \boldsymbol{p}_{k-1} \\ \rho_k \end{array}\right],$$

其中 $\rho_k = (\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{r}_0 - \mu_{k-1}d_{k-1}\rho_{k-1})/d_k$. 因此

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{C}_k\boldsymbol{p}_k = \boldsymbol{x}_0 + \boldsymbol{C}_{k-1}\boldsymbol{p}_{k-1} + \rho_k\boldsymbol{c}_k = \boldsymbol{x}_{k-1} + \rho_k\boldsymbol{c}_k.$$

这正好是我们所需要的关于 $\boldsymbol{x}_k$ 的迭代公式. 结合 (9.3.6) 和 (9.3.7), 我们就可用最小的工作量和存储量完成从 $(\boldsymbol{q}_{k-1}, \boldsymbol{c}_{k-1}, \boldsymbol{x}_{k-1})$ 到 $(\boldsymbol{q}_k, \boldsymbol{c}_k, \boldsymbol{x}_k)$ 的转变.

如果我们取 $q_1$ 为在初始余量 $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$ 方向上的单位向量, 则可得到更简明的结果. 当初始 Lanczos 向量这样选取时, 有 $\boldsymbol{q}_k^{\mathrm{T}}\boldsymbol{r}_0 = 0, k \geqslant 2$. 从 (9.3.3) 可知

$$\begin{aligned} \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k &= \boldsymbol{b} - \boldsymbol{A}(x_0 + \boldsymbol{Q}_k\boldsymbol{y}_k) \\ &= \boldsymbol{r}_0 - (\boldsymbol{Q}_k\boldsymbol{T}_k + \boldsymbol{r}_k\boldsymbol{e}_k^{\mathrm{T}})\boldsymbol{y}_k \\ &= \boldsymbol{r}_0 - \boldsymbol{Q}_k\boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{r}_0 - r_k\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{y}_k \\ &= -\boldsymbol{r}_k\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{y}_k. \end{aligned}$$

因此, 如果在 Lanczos 算法的某步中, 有 $\beta_k = \|\boldsymbol{r}_k\|_2 = 0$ 成立, 则 $\boldsymbol{A}\boldsymbol{x}_k = \boldsymbol{b}$. 此外, $\|\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}\|_2 = \beta_k|\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{y}_k|$, 故在迭代的同时, 可得到当前残量的估计. 综上所述, 我们有如下算法.

**算法 9.3.1** 矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 对称正定, $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{x}_0 \in \mathbb{R}^n$ 为初始向量 $(\boldsymbol{A}\boldsymbol{x}_0 \approx \boldsymbol{b})$, 则本算法计算出 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的解.

$r_0 = b - Ax_0$
$\beta_0 = \|r_0\|_2$
$q_0 = 0$
$k = 0$
**while** $\beta_k \neq 0$
  $q_{k+1} = r_k/\beta_k$
  $k = k + 1$
  $\alpha_k = q_k^{\mathrm{T}}Aq_k$

$r_k = (A - \alpha_k I)q_k - \beta_{k-1}q_{k-1}$

$\beta_k = \|r_k\|_2$

**if** $k = 1$

$d_1 = \alpha_1$

$c_1 = q_1$

$\beta_1 = \beta_0/\alpha_1$

$x_1 = \rho_1 q_1$

**else**

$\mu_{k-1} = \beta_{k-1}/d_{k-1}$

$d_k = \alpha_k - \beta_{k-1}\mu_{k-1}$

$c_k = q_k - \mu_{k-1}c_{k-1}$

$\rho_k = -\mu_{k-1}d_{k-1}\rho_{k-1}/d_k$

$x_k = x_{k-1} + \rho_k c_k$

**end**

**end**

$x = x_k$

本算法每次迭代需一个矩阵与向量相乘和几个 saxpy 运算. 它的数值表现将在下一章讨论, 那里它重新被推导且成为众所周知的**共轭梯度法**.

### 9.3.2 对称非定方程组

上节中的算法一个关键点是求三对角矩阵 $\boldsymbol{T}_k$ 的 $\text{LDL}^{\text{T}}$ 分解. 不幸的是, 当 $\boldsymbol{A}$(因而 $\boldsymbol{T}_k$) 非正定时, 这个分解是不稳定的. Paige and Saunders(1975) 建议用 $\boldsymbol{T}_k$ 的 "LQ" 分解来构造 $\boldsymbol{x}_k$ 的递推公式. 确切地说, 在第 $k$ 步, 我们有 Givens 旋转 $\boldsymbol{J}_1, \cdots, \boldsymbol{J}_{k-1}$, 使得

$$\boldsymbol{T}_k\boldsymbol{J}_1\cdots\boldsymbol{J}_{k-1} = \boldsymbol{L}_k = \begin{bmatrix} d_1 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ e_1 & d_2 & 0 & \cdots & \cdots & \cdots & 0 \\ f_1 & e_2 & d_3 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & f_{k-2} & e_{k-1} & d_k \end{bmatrix}.$$

注意, 利用这一分解, $\boldsymbol{x}_k$ 可表示为

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{Q}_k\boldsymbol{y}_k = \boldsymbol{Q}_k\boldsymbol{T}_k^{-1}\boldsymbol{Q}_k^{\text{T}}\boldsymbol{b} = \boldsymbol{W}_k\boldsymbol{s}_k,$$

其中 $\boldsymbol{W}_k = \boldsymbol{Q}_k\boldsymbol{J}_1\cdots\boldsymbol{J}_{k-1} \in \mathbb{R}^{n\times k}$, $\boldsymbol{s}_k \in \mathbb{R}^k$ 为方程组 $\boldsymbol{L}_k\boldsymbol{s}_k = \boldsymbol{Q}_k^{\text{T}}\boldsymbol{b}$ 的解. 观察这些方程, 从 $\boldsymbol{x}_{k-1}$ 以及 $\boldsymbol{W}_k$ 的最后一列 $\boldsymbol{w}_k$ 的一个容易计算的倍数, 可得出计算 $\boldsymbol{x}_k$ 的公式. 这就是 Paige and Saunders(1975) 提出的 SYMMLQ 方法.

另外一种途径是从 (9.3.3) 和定义 $\beta_k \boldsymbol{q}_{k+1} = \boldsymbol{r}_k$ 有

$$\boldsymbol{A}\boldsymbol{Q}_k = \boldsymbol{Q}_k\boldsymbol{T}_k + \beta_k \boldsymbol{q}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} = \boldsymbol{Q}_{k+1}\boldsymbol{H}_k,$$

其中

$$\boldsymbol{H}_k = \begin{bmatrix} \boldsymbol{T}_k \\ \beta_k \boldsymbol{e}_k^{\mathrm{T}} \end{bmatrix}.$$

这是 $(k+1)\times k$ 的上 Hessenberg 矩阵, 在 Paige and Saunders(1975) 的 MINRES 方法中起重要作用. 在该技巧中, $x_k$ 在集合 $\boldsymbol{x}_0+\mathrm{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}$ 上极小化 $\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2$. 注意

$$\begin{aligned}\|\boldsymbol{A}(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y})-\boldsymbol{b}\|_2 &= \|\boldsymbol{A}\boldsymbol{Q}_k\boldsymbol{y}-(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0)\|_2 \\ &= \|\boldsymbol{Q}_{k+1}\boldsymbol{H}_k\boldsymbol{y}-(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0)\|_2 = \|\boldsymbol{H}_k\boldsymbol{y}-\beta_0\boldsymbol{e}_1\|_2,\end{aligned}$$

这里假定了 $\boldsymbol{q}_1 = (\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_0)/\beta_0$ 为单位向量. 与 SYMMLQ 方法一样, 能导出一些递推公式从而有效地从 $\boldsymbol{x}_{k-1}$ 计算 $\boldsymbol{x}_k$. 这涉及 $\boldsymbol{H}_k$ 的 QR 分解.

共轭梯度法的表现将在下一章详细地讨论. SYMMLQ 方法和 MINRES 方法的收敛性更为复杂, 其讨论见 Paige, Parlett, and Van Detr Vorst(1995),

### 9.3.3 双对角化和 SVD

假定 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{B}$ 为矩阵 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 的双对角化, 其中

$$\boldsymbol{U} = [u_1,\cdots,u_m],\quad \boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{I}_m,$$
$$\boldsymbol{V} = [v_1,\cdots,v_n],\quad \boldsymbol{V}^{\mathrm{T}}\boldsymbol{V} = \boldsymbol{I}_n,$$

$$\boldsymbol{B} = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ 0 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \beta_{n-1} \\ 0 & \cdots & & 0 & \alpha_n \end{bmatrix}. \tag{9.3.8}$$

回忆 5.4.3 节中的内容, 此分解可用 Householder 变换计算, 它是 SVD 算法的前端.

不幸的是, 当 $\boldsymbol{A}$ 为大型稀疏矩阵时, 在 Householder 双对角化过程中, 会出现大型稠密子矩阵. 因此, 如果能发展一种不必对 $\boldsymbol{A}$ 做任何正交变换的直接计算 $\boldsymbol{B}$ 的方法, 将是非常完美的事情.

和 9.1.2 节中的做法一样, 对 $k=1:n$, 比较矩阵方程 $\boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{B}$ 与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{B}^{\mathrm{T}}$ 两边的列, 可得

$$\begin{aligned} \boldsymbol{A}\boldsymbol{v}_k &= \alpha_k\boldsymbol{u}_k + \beta_{k-1}\boldsymbol{u}_{k-1}, \quad \beta_0\boldsymbol{u}_0 \equiv \boldsymbol{0}, \\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_k &= \alpha_k\boldsymbol{v}_k + \beta_k\boldsymbol{v}_{k+1}, \quad \beta_n\boldsymbol{v}_{n+1} \equiv \boldsymbol{0}. \end{aligned} \tag{9.3.9}$$

令 $\boldsymbol{r}_k=\boldsymbol{A}\boldsymbol{v}_k-\beta_{k-1}\boldsymbol{v}_{k-1}, \boldsymbol{p}_k=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_k-\alpha_k\boldsymbol{v}_k$. 由单位正交性可得到 $\alpha_k=\pm\|\boldsymbol{r}_k\|_2, \boldsymbol{u}_k=\boldsymbol{r}_k/\alpha_k$, $\beta_k=\pm\|\boldsymbol{p}_k\|_2$ 和 $\boldsymbol{v}_{k+1}=\boldsymbol{p}_k/\beta_k$. 恰当地安排它们的顺序, 可得到双对角化长方形矩阵的 Lanczos 方法.

**给定** $n$ **维单位向量** ($\|\cdot\|_2$**范数**)$v_1$;

$p_0=v_1; \beta_0=1; k=0; u_0=0$

**while** $\beta_k \neq 0$

  $v_{k+1}=p_k/\beta_k$

  $k=k+1$

  $r_k=Av_k-\beta_{k-1}u_{k-1}$

  $\alpha_k=\|r_k\|_2$

  $u_k=r_k/\alpha_k$

  $p_k=A^{\mathrm{T}}u_k-\alpha_k v_k$

  $\beta_k=\|p_k\|_2$

**end**

如果 rank($\boldsymbol{A}$)=$n$, 则我们可以保证在 $\alpha_k$ 中不会有零出现. 实际上, 若 $\alpha_k=0$, 则 $\mathrm{span}\{\boldsymbol{A}\boldsymbol{v}_1,\cdots,\boldsymbol{A}\boldsymbol{v}_k\}\subset\mathrm{span}\{\boldsymbol{u}_1,\cdots,\boldsymbol{u}_{k-1}\}$, 这说明 $\boldsymbol{A}$ 是秩亏损的.

若 $\beta_k=0$, 则不难证明:

$$\begin{aligned}\boldsymbol{A}[\boldsymbol{v}_1,\cdots,\boldsymbol{v}_k]&=[\boldsymbol{u}_1,\cdots,\boldsymbol{u}_k]\boldsymbol{B}_k,\\ \boldsymbol{A}^{\mathrm{T}}[\boldsymbol{u}_1,\cdots,\boldsymbol{u}_k]&=[\boldsymbol{v}_1,\cdots,\boldsymbol{v}_k]\boldsymbol{B}_k^{\mathrm{T}},\end{aligned}$$

其中 $\boldsymbol{B}_k=B(1:k,1:k), \boldsymbol{B}$ 为 (9.3.8) 中所描述的矩阵. 因此, $u$ 和 $v$ 为奇异向量, 且 $\sigma(\boldsymbol{B}_k)\subset\sigma(\boldsymbol{A})$. Paige(1974) 讨论了 Lanczos 双对角化. 也可见 Cullum and Willoughby(1985a, 1985b). 在本质上, 它等价于把 Lanczos 三对角化技巧作用于对称矩阵

$$\boldsymbol{C}=\begin{bmatrix}\boldsymbol{0} & \boldsymbol{A}\\ \boldsymbol{A}^{\mathrm{T}} & \boldsymbol{0}\end{bmatrix}.$$

我们在 8.6 节开始时就证明了 $\lambda_i(\boldsymbol{C})=\sigma_i(\boldsymbol{A})=-\lambda_{n+m-i+1}(\boldsymbol{C}), i=1:n$. 由于这一点, 双对角化矩阵的大奇异值是 $\boldsymbol{A}$ 的大奇异值的很好近似也就不足为奇了. $\boldsymbol{C}$ 的内部特征值对应于 $\boldsymbol{A}$ 的小奇异值, 但这没有很好地近似. 与 Kaniel-Paige 理论相当的关于 Lanczos 双对角化的定理, 见 Luk(1978) 和 Colub, Luk, and Overton(1981). 前两节中的分析、算法和数值方面的讨论都可自然地平移到双对角化.

### 9.3.4 最小二乘问题

满秩的最小二乘问题 $\min\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2$ 可以通过双对角化来求解. 特别地,

$$\boldsymbol{x}_{\mathrm{LS}}=\boldsymbol{V}\boldsymbol{y}_{\mathrm{LS}}=\sum_{i=1}^{n}y_i v_i,$$

其中 $\boldsymbol{y} = [y_1, \cdots, y_n]^{\mathrm{T}}$ 为方程 $\boldsymbol{B}\boldsymbol{y} = [\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{b}, \cdots, \boldsymbol{u}_n^{\mathrm{T}}\boldsymbol{b}]^{\mathrm{T}}$ 的解. 注意, 因为 $\boldsymbol{B}$ 是上双对角矩阵, 所以只有当 $\boldsymbol{B}$ 的双对角化完成之后才能求解 $\boldsymbol{y}$. 此外, 还必须存储向量 $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n$, 当 $n$ 很大时, 这是令人不快的情形.

如果将 $\boldsymbol{A}$ 化成下双对角形式:

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{B} = \left[\begin{array}{ccccc} \alpha_1 & 0 & \cdots & \cdots & 0 \\ \beta_1 & \alpha_2 & \cdots & \cdots & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & & \ddots & \alpha_n \\ 0 & \cdots & \cdots & 0 & \beta_n \\ \hline & & \mathbf{0} & & \end{array}\right],$$

基于双对角化的稀疏最小二乘算法之构造就能很方便地完成. 这里 $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n]$ 和 $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m]$ 是正交矩阵. 比较方程 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{B}^{\mathrm{T}}$ 与 $\boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{B}$ 两边的列, 可得

$$\begin{aligned} \boldsymbol{A}^{\mathrm{T}}\boldsymbol{u}_k &= \beta_{k-1}\boldsymbol{v}_{k-1} + \alpha_k\boldsymbol{v}_k, \quad \beta_0\boldsymbol{v}_0 \equiv \mathbf{0}, \\ \boldsymbol{A}\boldsymbol{v}_k &= \alpha_k\boldsymbol{u}_k + \beta_k\boldsymbol{u}_{k+1}. \end{aligned}$$

从这些方程可直接给出一个 Lanczos 算法, 它与 (9.3.10) 很相似, 只是 $\boldsymbol{u}_1$ 为初始向量.

令 $\boldsymbol{V}_k = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k], \boldsymbol{U}_k = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k], \boldsymbol{B}_k = \boldsymbol{B}(1:k+1, 1:k)$, 则 $\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{U}_{k+1}\boldsymbol{B}_k$. 我们的目标是计算 $\boldsymbol{x}_k$, 使得 $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$ 在所有形为 $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{y}$ 的向量集上达到极小, 这里 $\boldsymbol{y} \in \mathbb{R}^k$ 和 $\boldsymbol{x}_0 \in \mathbb{R}^n$ 为初始向量. 若令 $\boldsymbol{u}_1 = (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)/\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0\|_2$, 则有

$$\boldsymbol{A}(\boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{y}) - \boldsymbol{b} = \boldsymbol{U}_{k+1}\boldsymbol{B}_k\boldsymbol{y} - \beta_1\boldsymbol{U}_{k+1}\boldsymbol{e}_1 = \boldsymbol{U}_{k+1}(\boldsymbol{B}_k\boldsymbol{y} - \beta_1\boldsymbol{e}_1),$$

其中 $\boldsymbol{e}_1 = I_{k+1}(:, 1)$. 由此可知, 若 $\boldsymbol{y}_k$ 为 $(k+1) \times k$ 下双对角最小二乘问题

$$\min \|\boldsymbol{B}_{k+1}\boldsymbol{y} - \beta_1\boldsymbol{e}_1\|_2$$

的解, 则 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{y}_k$. 既然 $\boldsymbol{B}_k$ 为下双对角矩阵, 容易计算 Givens 旋转 $\boldsymbol{J}_1, \boldsymbol{J}_2, \cdots, \boldsymbol{J}_k$, 使得

$$\boldsymbol{J}_k \cdots \boldsymbol{J}_1\boldsymbol{B}_k = \left[\begin{array}{c} \boldsymbol{R}_k \\ \mathbf{0} \end{array}\right] \begin{array}{l} k \\ 1 \end{array}$$

为上双对角矩阵. 若 $\boldsymbol{J}_k \cdots \boldsymbol{J}_1\boldsymbol{U}_{k+1}^{\mathrm{T}}\boldsymbol{b} = \left[\begin{array}{c} d_k \\ u \end{array}\right] \begin{array}{l} k \\ 1 \end{array}$, 则可得 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{V}_k\boldsymbol{y}_k = \boldsymbol{W}_k\boldsymbol{d}_k$, 其中 $\boldsymbol{W}_k = \boldsymbol{V}_k\boldsymbol{R}_k^{-1}$. Paige and Saunders(1982a) 指出如何用一个简单的递

推从 $x_{k-1}$ 得到 $x_k$, 递推涉及 $\boldsymbol{W}_k$ 的最后一列. 这样得到一个称为 LSQR 的稀疏最小二乘算法, 这仅需要少数几个 $n$ 维向量的存储空间.

## 习 题

**9.3.1** 修改算法 9.3.1, 使得它能够对 9.3.2 节中的对称非定问题进行计算.

**9.3.2** 为了有效地进行 (9.3.10), 需多少向量存储空间?

**9.3.3** 假定 $\boldsymbol{A}$ 是秩亏损的, 且 (9.3.10) 中 $a_k = 0$. 怎样确定 $u_k$, 使得迭代能进行下去?

**9.3.4** 设计出 (9.3.10) 的下双对角形式, 并详细给出 9.3.4 节中所概括的最小二乘算法.

## 本节注释与参考文献

本节中很多材料取自下列文章:

C. C. Paige(1974). "Bidiagonalization of Matrices and Solution of Linear Equations," *SIAM J. Num. Anal.* 11, 197–209.

C. C. Paige and M. A. Saunders(1975). "Solution of Sparse Indefinite Systems of Linear Equations," *SIAM J. Num. Anal.* 12, 617–629.

C. C. Paige and M. A. Saunders(1982a). "LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares," *ACM Trans. Math. Soft.* 8, 43–71.

C. C. Paige and M. A. Saunders(1982b). "Algorithm 583 LSQR: Sparse Linear Equations and Least Squares Problems," *ACM Trans. Math. Soft.* 8, 195–209.

M. A. Sanders(1995). "Solution of Sparse Rectangular Systems," *BIT* 35, 588–604.

也可参见 Cullum and Willoughby(1985a, 1985b) 以及:

O. Widlund(1978). "A Lanczos Method for a Class of Nonsymmetric Systems of Linear Equations," *SIAM J. Numer. Anal.* 15, 801–812.

B. N. Parlett(1980). "A New Look at the Lanczos Algorithm for Solving Symmetric Systems of Linear Equations," *Lin. Alg. and Its Applic.* 29, 323–346.

G. H. Golub, F. T. Luk, and M. Overton(1981). "A Block Lanczos Method for Computing the Singular Values and Corresponding Singular Vectors of a Matrix," *ACM Trans. Math. Soft.* 7, 149–169.

J. Cullum, R. A. Willoughby, and M. Lake(1983). "A Lanczos Algorithm for Computing Singular Values and Vectors of Large Matrices," *SIAM J. Sci and Stat. Comp.* 4, 197–215.

Y. Saad(1987). "On the Lanczos Method for Solving Symmetric Systems with Several Right Hand Sides," *Math. Comp.* 48, 651–662.

M. Berry and G. H. Golub(1991). "Estimating the Largest Singular Values of Large Sparse Matrices via Modified Moments," *Numerical Algorithms* 1, 353–374.

C. C. Paige, B. N. Parlett, and H. A. Van Der Vorst(1995). "Approximate Solutions and Eigenvalue Bounds from Krylov Subspaces," *Numer. Linear Algebra with Applic.* 2, 115–134.

# 9.4 Arnoldi 方法与非对称 Lanczos 方法

如果矩阵 $\boldsymbol{A}$ 非对称, 则正交三对角化 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 一般不存在. 这时有两种方法来处理. 一种是 Arnoldi 方法的, 其基本思想是一列一列地产生正交矩阵 $\boldsymbol{Q}$, 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ 为 Hessenberg 约化, 见 7.4 节. 另一种方法是非对称 Lanczos 方法, 它计算矩阵 $\boldsymbol{Q}=[\boldsymbol{q}_1,\cdots,\boldsymbol{q}_n]$ 和 $\boldsymbol{P}=[\boldsymbol{p}_1,\boldsymbol{p}_2,\cdots,\boldsymbol{p}_n]$ 的列, 使 $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}$ 是三对角矩阵且 $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{Q}=\boldsymbol{I}_n$. 对于大型非对称稀疏矩阵的特征问题, 这两种方法都很有效, 它们也都可用来解非对称稀疏的方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$. (见 10.4 节.)

### 9.4.1 基本的 Arnoldi 迭代

把 Lanczos 方法推广到非对称矩阵的一种方式归功于 Arnoldi(1951), 它利用 Hessenberg 约化 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$. 特别地, 设 $\boldsymbol{Q}=[\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_n]$, 比较方程 $\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{Q}\boldsymbol{H}$ 两边的列可得

$$\boldsymbol{A}\boldsymbol{q}_k=\sum_{i=1}^{k+1}h_{ik}\boldsymbol{q}_i,\quad 1\leqslant k\leqslant n-1.$$

把上述和式中的最后一项分离出来可得

$$h_{k+1,k}\boldsymbol{q}_{k+1}=\boldsymbol{A}\boldsymbol{q}_k-\sum_{i=1}^{k}h_{ik}\boldsymbol{q}_i:\equiv r_k,$$

其中 $h_{ik}=\boldsymbol{q}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_k, i=1:k$. 由此可知, 若 $\boldsymbol{r}_k\neq\boldsymbol{0}$, 则 $\boldsymbol{q}_{k+1}$ 可通过下式来确定:

$$\boldsymbol{q}_{k+1}=\boldsymbol{r}_k/h_{k+1},k,$$

其中 $h_{k+1,k}=\|\boldsymbol{r}_k\|_2$. 这些方程定义了 Arnoldi 算法, 与对称的 Lanczos 算法 (9.1.3) 类似, 我们得到

$$
\begin{aligned}
&r_0=q_1\\
&h_{10}=1\\
&k=0\\
&\textbf{while}(h_{k+1},k\neq 0)\\
&\qquad q_{k+1}=r_k/h_{k+1,k}\\
&\qquad k=k+1\\
&\qquad r_k=Aq_k\\
&\qquad \textbf{for } i=1:k\\
&\qquad\qquad h_{ik}=q_i^{\mathrm{T}}w\\
&\qquad\qquad r_k=r_k-h_{ik}q_i\\
&\qquad \textbf{end}\\
&\qquad h_{k+1,k}=\|r_k\|_2
\end{aligned}
\tag{9.4.1}
$$

**end**

我们假定 $\boldsymbol{q}_1$ 为给定的 2 范数单位初始向量. $\boldsymbol{q}_k$ 称为 Arnoldi 向量, 它们构成了 Krylov 子空间 $\boldsymbol{K}(\boldsymbol{A},\boldsymbol{q}_1,k)$ 的一组标准正交基:

$$\text{span}\{\boldsymbol{q}_1,\cdots,\boldsymbol{q}_k\}=\text{span}\{\boldsymbol{q}_1,\boldsymbol{A}\boldsymbol{q}_1,\cdots,\boldsymbol{A}^{k-1}\boldsymbol{q}_1\}. \tag{9.4.2}$$

迭代 $k$ 步以后, 算法的运行情况可用第 $k$ 步 Arnoldi 分解来概括:

$$\boldsymbol{A}\boldsymbol{Q}_k=\boldsymbol{Q}_k\boldsymbol{H}_k+r_k\boldsymbol{e}_k^{\mathrm{T}}, \tag{9.4.3}$$

其中 $\boldsymbol{Q}_k=[\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_k],\boldsymbol{e}_k=I_k(:,k)$,

$$\boldsymbol{H}_k=\begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2k} \\ 0 & h_{32} & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & h_{k,k-1} & h_{kk} \end{bmatrix}.$$

如果 $\boldsymbol{r}_k=\boldsymbol{0}$, 则 $\boldsymbol{Q}_k$ 的列组成了一不变子空间, 且 $\lambda(\boldsymbol{H}_k)\subseteq\lambda(\boldsymbol{A})$. 否则, 问题的焦点是如何从 Hessenberg 矩阵 $\boldsymbol{H}_k$ 和 Arnoldi 向量组成的矩阵 $\boldsymbol{Q}_k$ 中提取有关 $\boldsymbol{A}$ 的特征信息.

若 $\boldsymbol{y}\in\mathbb{R}^k$ 为 $\boldsymbol{H}_k$ 的 2 范数单位特征向量, $\boldsymbol{H}_k\boldsymbol{y}=\lambda\boldsymbol{y}$, 则从 (9.4.3) 可得

$$(\boldsymbol{A}-\lambda\boldsymbol{I})\boldsymbol{x}=(\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{y})r_k,$$

其中 $\boldsymbol{x}=\boldsymbol{Q}_k\boldsymbol{y}$. 我们称 $\lambda$ 为 Ritz 值, $x$ 为相应的 Ritz 向量. 数 $|\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{y}|\|\boldsymbol{r}_k\|_2$ 的大小可用来衡量误差界, 虽然相关的扰动理论不像对称矩阵的情况那样有章可循.

Wilkinson(1965,382 页) 讨论了 Arnoldi 迭代的一些数值上的性质. 和对称矩阵的 Lanczos 迭代一样, 向量 $\boldsymbol{q}_i$ 间的正交性也会丢失. 要得到实用的 Arnoldi 特征值算法, 必须注意 (9.4.1) 的两个其他特征:

- *在第 $k$ 步要用到 Arnoldi 向量 $\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_k$ 且 $H_k(1:k,k)$ 的计算需要 $O(kn)$ 个 flop. 因此, 当产生一长串 Arnoldi 向量时, 要付出很高的代价.*
- *$\boldsymbol{H}_k$ 的特征值并不按照 Kaniel 和 Paige 方式近似 $\boldsymbol{A}$ 的特征值. 这与对称时能很快显露 $\boldsymbol{A}$ 的两端的特征值信息形成了鲜明的对比. 在 Arnoldi 方法中, 能否较早得到特征值的信息强烈依赖于 $\boldsymbol{q}_1$ 的选取.*

这些事实表明, 在使用 Arnoldi 方法时, 应当反复地、仔细地选择重新开始以及控制最大迭代迭代. (回想 9.2.7 节中的 $s$ 步 Lanczos 方法.)

### 9.4.2 重开始的 Arnoldi 方法

考虑 Arnoldi 算法运行 $m$ 步后, 从 Arnoldi 向量 $\boldsymbol{q}_1,\cdots,\boldsymbol{q}_m$ 的生成空间中选取一向量 $\boldsymbol{q}_+$, 再重新运行 Arnoldi 算法. 由关系式 (9.4.2) 知 $\boldsymbol{q}_+$ 具有表达式:

$$q_+ = p(A)q_1,$$

其中 $p(x)$ 为某个不高于 $m-1$ 次的多项式. 如果 $Av_i = \lambda_i v_i, i = 1:n, q_1$ 有特征向量展开式:

$$q_1 = a_1 v_1 + \cdots + a_n v_n,$$

则 $q_+ = a_1 p(\lambda_1) v_1 + \cdots + a_n p(\lambda_n) v_n$. 注意, 空间 $K(A, q_+, m)$ 在 $p(\lambda)$ 强调的特征向量上占优势. 也就是说, 如果 $P(\lambda_{\text{wanted}})$ 比 $P(\lambda_{\text{unwanted}})$ 要大, 则 Krylov 空间 $K(A, q_+, m)$ 对特征向量 $x_{\text{wanted}}$ 的近似比对 $x_{\text{unwanted}}$ 的近似要好得多. (可以用 Schur 向量和不变子空间来进行这一讨论, 而不必涉及特定的特征向量.)

因此, 从 $K(A, q_1, m)$ 中选取一个好的重开始向量 $q_+$ 就是挑选一多项式"过滤器", 用来除去谱中不需要的部分. 有许多不同的实用方法是基于已计算的 Ritz 向量, 见 Saad(1980, 1984, 1992).

我们介绍 Sorensen(1992) 的一种方法, 它隐含地应用带位移的 QR 迭代来确定新的重开始向量. 每经 $m$ 步以后重新开始. 我们假设 $m > j$, 其中 $j$ 为所需要的特征值的个数. Arnoldi 长度参数 $m$ 的选取依赖于维数 $n$、正交性丢失的影响和机器的存储限制.

经 $m$ 步后, 有 Arnoldi 分解

$$AQ_c = Q_c H_c + r_c e_m^{\mathrm{T}},$$

其中 $Q_c \in \mathbb{R}^{n \times m}$ 的列正交, $H_c \in \mathbb{R}^{n \times m}$ 为上 Hessenberg 矩阵且 $Q_c^{\mathrm{T}} r_c = 0$. 这里下标"$c$"代表"当前 (current)". 把带位移的 QR 迭代应用于 $H_c$:

$H^{(1)} = H_c$
**for** $i = 1:p$
  $H^{(i)} - \mu_i I = V_i R_i$
  $H^{(i+1)} = R_i V_i + \mu_i I$
**end**
$H_+ = H^{(p+1)}$

这里 $p = m - j$, 并假设应用了 7.5.5 节中的隐含带位移的 QR 过程. 我们将简要地讨论位移的选取.

正交矩阵 $V = V_1 \cdots V_p$ 有三条重要的性质:

(1) $H_+ = V^{\mathrm{T}} H_c V$, 这是因为 $V_i^{\mathrm{T}} H^{(i)} V_i = H^{(i+1)}$.

(2) $[V]_{mi} = 0, i = 1:j-1$. 这是因为每个 $V_i$ 都是上 Hessenberg 矩阵, 从而 $V \in \mathbb{R}^{m \times m}$ 有下带宽 $p = m - j$.

(3) $V$ 的第一列有如下形式:

$$Ve_1 = \alpha(H_c - \mu_p I)(H_c - \mu_{p-1} I) \cdots (H_c - \mu_1 I) e_1, \tag{9.4.4}$$

其中 $\alpha$ 为标量.

为了验证性质 (3), 只需考虑 $p=2$ 的情形:

$$\begin{aligned}\boldsymbol{V}\boldsymbol{R}_2\boldsymbol{R}_1 &= \boldsymbol{V}_1(\boldsymbol{V}_2\boldsymbol{R}_2)\boldsymbol{R}_1 = \boldsymbol{V}_1(\boldsymbol{H}^{(2)}-\mu_2\boldsymbol{I})\boldsymbol{R}_1\\ &= \boldsymbol{V}_1(\boldsymbol{V}_1^{\mathrm{T}}\boldsymbol{H}^{(1)}\boldsymbol{V}_1-\mu_2\boldsymbol{I})\boldsymbol{R}_1 = (\boldsymbol{H}^{(1)}-\mu_2\boldsymbol{I})\boldsymbol{V}_1\boldsymbol{R}_1\\ &= (\boldsymbol{H}^{(1)}-\mu_2\boldsymbol{I})(\boldsymbol{H}^{(1)}-\mu_1\boldsymbol{I}) = (\boldsymbol{H}_c-\mu_2\boldsymbol{I})(\boldsymbol{H}_c-\mu_1\boldsymbol{I}).\end{aligned}$$

因为 $\boldsymbol{R}_2\boldsymbol{R}_1$ 为上三角形矩阵, 矩阵 $\boldsymbol{V}=\boldsymbol{V}_1\boldsymbol{V}_2$ 的第一列为 $(\boldsymbol{H}_c-\mu_2\boldsymbol{I})(\boldsymbol{H}_c-\mu_1\boldsymbol{I})$ 的第一列的倍数.

现在, 我们说明如何利用矩阵 $\boldsymbol{V}$ 隐含地选取新的初始向量来重开始 Arnoldi 过程. 从性质 (1) 可得 (9.4.3) 的变换形式:

$$\boldsymbol{A}\boldsymbol{Q}_+ = \boldsymbol{Q}_+\boldsymbol{H}_+ + r_c\boldsymbol{e}_m^{\mathrm{T}}\boldsymbol{V},$$

其中 $\boldsymbol{Q}_+=\boldsymbol{Q}_c\boldsymbol{V}$. 这并不是一个长度为 $m$ 的新的 Arnoldi 分解, 因为 $e_m^{\mathrm{T}}V$ 并不是 $e_m^{\mathrm{T}}$ 的倍数. 然而由性质 (2) 有

$$AQ_+(:,1:j) = Q_+(:,1:j)H_+(1:j,1:j) + v_{mj}r_ce_j^{\mathrm{T}}, \tag{9.4.5}$$

这是一个长度为 $j$ 的 Arnoldi 分解. 在 $j+1$ 步“跳到”基本的 Arnoldi 迭代并运行 $p$ 步. 我们可以把 (9.4.4) 延伸为一个新的长度为 $m$ 的 Arnoldi 分解. 此外, 利用性质 (3), 相应的初始向量 $\boldsymbol{q}_1^{(new)}=Q_+(:,1)$ 有下述特征:

$$\begin{aligned}Q_+(:,1) = \boldsymbol{Q}_c\boldsymbol{V}\boldsymbol{e}_1 &= \alpha\boldsymbol{Q}_c(\boldsymbol{H}_c-\mu_p\boldsymbol{I})\cdots(\boldsymbol{H}_c-\mu_1\boldsymbol{I})\boldsymbol{e}_1\\ &= \alpha(\boldsymbol{A}-\mu_p\boldsymbol{I})\cdots(\boldsymbol{A}-\mu_1\boldsymbol{I})\boldsymbol{Q}_c\boldsymbol{e}_1.\end{aligned} \tag{9.4.6}$$

最后一步用到了等式

$$(\boldsymbol{A}-\mu\boldsymbol{I})\boldsymbol{Q}_c = \boldsymbol{Q}_c(\boldsymbol{H}_c-\mu\boldsymbol{I}) + \boldsymbol{r}\boldsymbol{e}_m^{\mathrm{T}}$$

以及 $\boldsymbol{e}_m^{\mathrm{T}}f(\boldsymbol{H}_c)\boldsymbol{e}_1=0$ 对任意不高于 $p-1$ 次的多项式成立.

因此, $\boldsymbol{q}_1^{(new)}=p(\boldsymbol{A})\boldsymbol{q}_1$, 其中 $p(\lambda)$ 是多项式

$$p(\lambda) = (\lambda-\mu_1)(\lambda-\mu_2)\cdots(\lambda-\mu_p).$$

这表明这些位移是“过滤”多项式的零点. 位移的一种有趣的选择是计算 $\lambda(\boldsymbol{H}_c)$, 且确定有用的特征值 $\widetilde{\lambda}_1,\widetilde{\lambda}_2,\cdots,\widetilde{\lambda}_j$,

$$\lambda(\boldsymbol{H}_c) = \{\widetilde{\lambda}_1,\cdots,\widetilde{\lambda}_j\}\cup\{\widetilde{\lambda}_{j+1},\cdots,\widetilde{\lambda}_m\}.$$

令 $\mu_i=\widetilde{\lambda}_{i+j}, i=1:p$, 这就产生了一个“过滤”多项式, 降低了谱中不需要的部分之影响.

这里我们只介绍了**隐含重开始的**Arnoldi**方法**的非常基本的部分. 该方法有许多吸引人的特性. 实现细节和进一步分析见 Lehoucq and Sorensen(1996) 以及 Morgan(1996).

### 9.4.3 非对称 Lanczos 三角化

另一种推广对称 Lanczos 方法的做法是用一般的相似变换将 $\boldsymbol{A}$ 化为三对角形式. 假定 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, 存在非奇异矩阵 $\boldsymbol{Q}$ 使得

$$\boldsymbol{Q}^{-1}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{T} = \begin{bmatrix} \alpha_1 & \gamma_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \gamma_{n-1} \\ 0 & \cdots & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

利用如下列划分:

$$\begin{aligned} \boldsymbol{Q} &= [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n], \\ \boldsymbol{Q}^{-\mathrm{T}} &= \boldsymbol{P} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n], \end{aligned}$$

比较方程 $\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{T}$ 与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} = \boldsymbol{P}\boldsymbol{T}^{\mathrm{T}}$ 两边的各列, 我们发现: 对于 $k = 1 : n-1$,

$$\begin{aligned} \boldsymbol{A}\boldsymbol{q}_k &= \gamma_{k-1}\boldsymbol{q}_{k-1} + \alpha_k\boldsymbol{q}_k + \beta_k\boldsymbol{q}_{k+1}, \qquad \gamma_0\boldsymbol{q}_0 \equiv \boldsymbol{0}, \\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k &= \beta_{k-1}\boldsymbol{p}_{k-1} + \alpha_k\boldsymbol{p}_k + \gamma_k\boldsymbol{p}_{k+1}, \quad \beta_0\boldsymbol{p}_0 \equiv \boldsymbol{0}. \end{aligned}$$

这些方程和**双正交性条件** $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{Q} = \boldsymbol{I}_n$ 意味着

$$\alpha_k = \boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{q}_k,$$

$$\begin{aligned} \beta_k\boldsymbol{q}_{k+1} &\equiv \boldsymbol{r}_k = (\boldsymbol{A} - \alpha_k\boldsymbol{I})\boldsymbol{q}_k - \gamma_{k-1}\boldsymbol{q}_{k-1}, \\ \gamma_k\boldsymbol{p}_{k+1} &\equiv \boldsymbol{s}_k = (\boldsymbol{A} - \alpha_k\boldsymbol{I})^{\mathrm{T}}\boldsymbol{p}_k - \beta_{k-1}\boldsymbol{p}_{k-1}. \end{aligned}$$

在选择数 $\beta_k$ 和 $\gamma_k$ 时有一定的自由度. 注意到

$$1 = \boldsymbol{p}_{k+1}^{\mathrm{T}}\boldsymbol{q}_{k+1} = (\boldsymbol{s}_k/\gamma_k)^{\mathrm{T}}(\boldsymbol{r}_k/\beta_k).$$

一旦 $\beta_k$ 确定了, 就有 $\gamma_k = \boldsymbol{s}_k^{\mathrm{T}}\boldsymbol{r}_k/\beta_k$. 选择 $\beta_k$ 为"标准形式": $\beta_k = \|\boldsymbol{r}_k\|_2$, 就得下列算法:

```
给定 2 范数单位向量 p_1, q_1, 且 p_1^T q_1 ≠ 0.
k = 0
q_0 = 0; r_0 = q_1
p_0 = 0; s_0 = p_1
while (r_k ≠ 0) ∧ (s_k ≠ 0) ∧ (s_k^T r_k ≠ 0)
    β_k = ||r_k||_2
    γ_k = s_k^T r_k / β_k
    q_{k+1} = r_k / β_k
    p_{k+1} = s_k / γ_k
```

$$
\begin{aligned}
&k = k+1 \\
&\alpha_k = p_k^{\mathrm{T}} A q_k \\
&r_k = (A - \alpha_k I) q_k - \gamma_{k-1} q_{k-1} \\
&s_k = (A - \alpha_k I)^T p_k - \beta_{k-1} p_{k-1}
\end{aligned}
\tag{9.4.7}
$$

**end**

如果令

$$
\boldsymbol{T}_k = \begin{bmatrix} \alpha_1 & \gamma_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \gamma_{k-1} \\ 0 & \cdots & & \beta_{k-1} & \alpha_k \end{bmatrix},
$$

则在上述循环的底部, 情况可概括为以下方程:

$$
\boldsymbol{A}[\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k] = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k]\boldsymbol{T}_k + \boldsymbol{r}_k \boldsymbol{e}_k^{\mathrm{T}}, \tag{9.4.8}
$$

$$
\boldsymbol{A}^{\mathrm{T}}[\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k] = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k]\boldsymbol{T}_k^{\mathrm{T}} + \boldsymbol{s}_k \boldsymbol{e}_k^{\mathrm{T}}. \tag{9.4.9}
$$

若 $r_k = 0$, 则循环结束, $\mathrm{span}\{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_k\}$ 为 $\boldsymbol{A}$ 的一不变子空间. 若 $\boldsymbol{s}_k = \boldsymbol{0}$, 则循环也结束, $\mathrm{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}$ 为 $\boldsymbol{A}^{\mathrm{T}}$ 的一不变子空间. 然而, 当这些条件不成立且 $\boldsymbol{s}_k^{\mathrm{T}} \boldsymbol{r}_k = 0$ 时, 三对角化过程将终止, 但得不到任何不变子空间的信息. 这种情况称为*严重失败*. 关于此问题早期的讨论见 Wilkinson(1965, 389 页).

### 9.4.4 “向前看”的技巧

研究一下算法 (9.4.7) 的块形式中的严重失败现象是很有趣的. 为简单起见, 假定 $\boldsymbol{A} \in \mathbb{R}^{n \times n}, n = rp$. 考虑分解

$$
\boldsymbol{P}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{M}_1 & \boldsymbol{C}_1^{\mathrm{T}} & & \cdots & \boldsymbol{0} \\ \boldsymbol{B}_1 & \boldsymbol{M}_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \boldsymbol{C}_{r-1}^{\mathrm{T}} \\ \boldsymbol{0} & \cdots & & \boldsymbol{B}_{r-1} & \boldsymbol{M}_r \end{bmatrix}, \tag{9.4.10}
$$

其中每一块都是 $p \times p$ 阶子矩阵. 设 $\boldsymbol{Q} = [\boldsymbol{Q}_1, \boldsymbol{Q}_2, \cdots, \boldsymbol{Q}_r], \boldsymbol{P} = [\boldsymbol{P}_1, \boldsymbol{P}_2, \cdots, \boldsymbol{P}_r]$ 为 $\boldsymbol{Q}$ 和 $\boldsymbol{P}$ 恰当的分块形式. 比较 $\boldsymbol{AQ} = \boldsymbol{QT}$ 和 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{P} = \boldsymbol{P}\boldsymbol{T}^{\mathrm{T}}$ 两边相应的列块, 可得

$$
\begin{aligned}
\boldsymbol{Q}_{k+1}\boldsymbol{B}_k &= \boldsymbol{A}\boldsymbol{Q}_k - \boldsymbol{Q}_k\boldsymbol{M}_k - \boldsymbol{Q}_{k-1}\boldsymbol{C}_{k-1}^{\mathrm{T}} \equiv \boldsymbol{R}_k, \\
\boldsymbol{P}_{k+1}\boldsymbol{C}_k &= \boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{M}_k^{\mathrm{T}} - \boldsymbol{P}_{k-1}\boldsymbol{B}_{k-1}^{\mathrm{T}} \equiv \boldsymbol{S}_k.
\end{aligned}
$$

注意, $\boldsymbol{M}_k = \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k$. 若 $\boldsymbol{S}_k^{\mathrm{T}}\boldsymbol{R}_k \in \mathbb{R}^{p \times p}$ 非奇异, 我们计算 $\boldsymbol{B}_k, \boldsymbol{C}_k \in \mathbb{R}^{p \times p}$, 使得

$$C_k^{\mathrm{T}} B_k = S_k^{\mathrm{T}} R_k,$$

则

$$Q_{k+1} = R_k B_k^{-1} \tag{9.4.11}$$

$$P_{k+1} = S_k C_k^{-1} \tag{9.4.12}$$

满足 $P_{k+1}^{\mathrm{T}} Q_{k+1} = I_p$. 当 $S_k^{\mathrm{T}} R_k$ 奇异时, 就会出现严重失败现象.

一种解决 (9.4.7) 中严重失败问题的方法是寻找形如 (9.4.10) 的分解, 其中块的大小能动态地确定. 粗略地讲, 矩阵 $Q_{k+1}$ 和 $P_{k+1}$ 用一些特殊的递推关系一列一列地形成, 这些递推关系可计算非异矩阵 $P_{k+1}^{\mathrm{T}} Q_{k+1}$. 适当安排计算的顺序, 使得双正交性条件 $P_i^{\mathrm{T}} Q_{k+1} = 0$ 和 $Q_i^{\mathrm{T}} P_{k+1} = 0$ 对所有的 $i = 1:k$ 成立.

这种方法属于一种称为向前看Lanczos 方法族. 向前看步的长度是它产生 $Q_{k+1}$ 和 $P_{k+1}$ 的宽度. 如果这种宽度为 1, 则可利用传统的分块 Lanczos 步. Parlett, Taylor, and Liu(1985) 讨论了长度为 2 的向前步. 他们提出了"无法补救的失败"的概念. Freund, Gutknecht, and Nachtigal(1993) 概括了一般的情况, 并讨论了算法的许多细节. 考虑浮点运算时, 需处理"近似"严重失败的情况. 在实际计算中, 每个 $2 \times 2$ 的或更高阶的 $M_k$, 都相应于近似严重失败的情况.

## 习　题

**9.4.1** 证明 (9.4.1) 中的 Arnoldi 向量是相互正交的.

**9.4.2** 证明 (9.4.4).

**9.4.3** 证明 (9.4.6).

**9.4.4** 给出一个初始向量的例子, 使得非对称 Lanczos 迭代 (9.4.7) 中断, 且没得到任何不变子空间的信息. 利用矩阵

$$A = \begin{bmatrix} 1 & 6 & 2 \\ 3 & 0 & 2 \\ 1 & 3 & 5 \end{bmatrix}.$$

**9.4.5** 已知 $H \in \mathbb{R}^{n\times n}$ 是上 Hessenderg 矩阵, 试讨论如何计算一单位上三角形矩阵 $U$, 使得 $HU = UT$, 这里 $T$ 是三对角矩阵.

**9.4.6** 说明在非对称情形中, 求特征值的 QR 算法不能保持三对角结构.

## 本节注释与参考文献

关于 Arnoldi 迭代及其实现的文献包括 Saad(1992) 和

W. E. Arnoldi(1951). "The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem," *Quarterly of Applied Mathematics* 9, 17–29.

Y. Saad(1980). "Variations of Arnoldi's Method for Computing Eigenelements of Large Unsymmetric Matrices," *Lin. Alg. and Its Applic.* 34, 269–295.

Y. Saad(1984). "Chebyshev Acceleration Techniques for Solving Nonsymmetric Eigenvalue Problems," *Math. Comp.* 42, 567–588.

D. G. Sorensen(1992). "Implicit Application of Polynomial Filters in a k-Step Amoldi Method," *SIAM J. Matrix Anal. Appl.* 13, 357–385.

D. C. Sorensen(1995). "Implicitly Restarted Arnoldi/Lanczos Methods for Large Scale Eigenvalue Calculations," in *Proceedings of the ICASE/LaRC Workshop on Parallel Numerical Algorithms, May* 23–25, 1994, D. E. Keyes, A. Sameh, and V. Venkatakrishnan(eds), Kluwer.

R. B, Lehoucq(1995). "Analysis and Implementation of an Implicitly Restarted Arnoldi Iteration," Ph. D. thesis, Rice University, Houston Texas.

R. B. Lehoucq(1996). "Restarting an Arnoldi Reduction," Report MCS-P591-0496, Argonne National Laboratory, Argonne Illinois.

R. B. Lehoucq and D. C. Sorensen(1996). "Deflation Techniques for an Implicitly Restarted Iteration," *SIAM J. Matrix Analysis and Applic, to appear.*

R. B. Morgan(1996). "On Restarting the Arnoldi Method for Large Nonsymmetric Eigenvalue Problems," *Math Comp* 65, 1213–1230.

相关的文章有:

A. Ruhe(1984). "Rational Krylov Algorithms for Eigenvalue Computation," *Lin. Alg. and Its Applic.* 58, 391–405.

A. Ruhe(1994). "Rational Krylov Algorithms for Nonsymmetric Eigenvalue Problems II. Matrix Pairs," *Lin. Alg. and Its Applic.* 197, 283–295.

A. Ruhe(1994). "The Rational Krylov Algorithm for Nonsymmetric Eigenvalue Problems III: Complex Shifts for Real Matrices," *BIT* 34, 165–176.

T. Huckle(1994). "The Arnoldi Method for Normal Matrices," *SIAM J. Matrix Anal. Appl.* 15, 479–489.

C. C. Paige, B. N. Parlett, and H. A. Van Der Vorst(1995). "Approximate Solutions and Eigenvalue Bounds from Krylov Subspaces," *Numer. Linear Algebra with Applic.* 2, 115–134.

K. C. Toh and L. N. Trefethen(1996). "Calculation of Pseudospectra by the Arnoldi Iteration," *SIAM J. Sci. Comp.* 17, 1–15.

非对称 Lanczos 方法和相关的向前看技巧在下文中很好地给出:

B. N. Parlett, D. Taylor, and Z. Liu(1985). "A Look-Ahead Lanczos Algorithm for Unsymmetric Matrices," *Math. Comp.* 44, 105–125.

R. W. Freund, M. Gutknecht, and N. Nachtigal(1993). "An Implementation of the Look-Ahead Lanczos Algorithm for Non-Hermitian Matrices," *SIAM J. Sci. and Stat. Comp.* 14, 137–158.

也见:

Y. Saad(1982). "The Lanczos Biorthogonalization Algorithm and Other Oblique Projection Methods for Solving Large Unsymmetric Eigenproblems," *SIAM J. Numer. Anal.* 19, 485–506.

G. A. Geist(1991). "Reduction of a General Matrix to Tridiagonal Form," *SIAM J. Matrix Anal. Appl.* 12, 362–373.

C. Brezinski, M. Zaglia, and H. Sadok(1991). "Avoiding Breakdown and Near Breakdown in Lanczos Tyoe Algorithms," *Numer. Alg.* 1, 261–284.

S. K. Kim and A. T. Chronopoulos(1991). "A Class of Lanczos-Like Algorithms Implemented on Paralled Computers," *Parallel Comput.* 17, 763–778.

B. N. Parlett(1992). "Reduction to Tridiagonal Form and Minimal Realizations," *SIAM J. Matrix Anal. Appl.* 13, 567–593.

M. Gutknecht(1992). "A Completed Theory of the Unsymmetric Lanczos Process and Related Algorithms, Part I," *SIAM J. Matrix Anal. Appl.* 13, 594–639.

M. Gutknecht(1994). "A Completed Theory of the Unsymmetric Lanczos Process and Related Algorithms, Part II," *SIAM J. Matrix Anal. Appl.* 15, 15–58.

Z. Bai(1994). "Error Analysis of the Lanczos Algorithm for Nonsymmetric Eigenvalue Problem," *Math. Comp.* 62, 209–226.

T. Huckle(1995). "Low-Rank Modification of the Unsymmetric Lanczos Algorithm," *Math. Comp.* 64, 1577–1588.

Z. Jia(1995). "The Convergence of Generalized Lanczos Methods for Large Unsymmetric Eigenproblems," *SIAM J. Matrix Anal. Applic.* 16, 543–562.

M. T. Chu, R. E. Funderlic, and G. H. Golub(1995). "A Rank-One Reduction Formula and Its Applications to Matrix Factorizations," *SIAM Review* 37, 512–530.

H. A. Van der Vorst(1982). "A Generalized Lanczos Scheme," *Math. Comp.* 39, 559–562.

D. Boley and G. H. Golub(1984). "The Lanczos-Arnoldi Algorithm and Controllability," *Syst. Control Lett.* 4, 317–324.

# 第 10 章 线性方程组的迭代解法

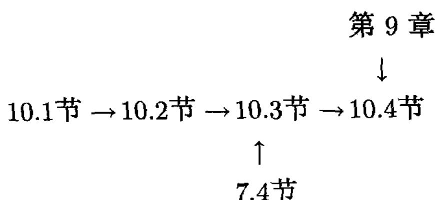
上一章, 我们介绍了如何用 Larczos 迭代解各种各样的线性方程组和最小二乘问题. 这些方法适用于大型稀疏问题, 因为它们不需要相应的矩阵分解. 本章将继续讨论具有这类性质的线性方程组的解法.

10.1 节粗略地介绍了一些经典的迭代: Jacobi 迭代、Gauss-Seidel 迭代、逐次超松弛迭代和切比雪夫半迭代等, 这些方法的介绍之所以很简要是因为我们本章的重点是阐述共轭梯度法, 在 10.2 节中, 我们从最速下降法自然地发展到这种重要的技巧, 回想一下, 在 9.3 节中, 我们在 Lanczos 迭代中就已提及了共轭梯度法. 这里又导出这一方法, 是因为它有一些实用的变化形式, 这便是 10.3 节的主要内容. 在 10.4 节, 我们将其推广到非对称的情形.

我们提醒读者, 本章中的一些记号不太一致. 在 10.1 节中, 所介绍的方法建立在"$(i,j)$ 层次", 故需用上标: $x_i^{(k)}$ 表示向量 $x^{(k)}$ 的第 $i$ 个分量. 在其他节中, 算法不需要显式利用矩阵 (向量) 的元素. 因此, 在 10.2 节 ~10.4 节中, 我们不用上标, 记向量序列为 $\{x_k\}$.

*预备知识*

阅读本章, 需要掌握第 1 章、2.1 节 ~2.5 节、2.7 节、第 3 章以及 4.1 节 ~4.3 节的知识. 本章各节间的关系如下:




关于迭代法的专著包括 Varga(1962), Young(1971), Hageman and Young(1981), 以及 Axelsson(1994). Barrett *et al* (1993) 给出软件的"模板"十分有用. 有时, 更偏向于用直接 (非迭代) 方法解大型稀疏问题. 见 George and Liu(1981) 以及 Duff, Erisman, and Reid(1986).

## 10.1 标准的迭代方法

在第 3 章和第 4 章中, 线性方程组的解法涉及系数矩阵 $A$ 的分解. 这类方法称为*直接法*. 当 $A$ 是大型稀疏矩阵时, 直接法不可行, 因为所求的分解因子是稠密

的. 当 $\boldsymbol{A}$ 是带状时 (参见 4.3 节), 是一个例外. 然而, 在许多带状矩阵问题中, 甚至带本身也是稀疏的, 使得许多诸如带状 Cholesky 分解等算法也难以实现.

对稀疏线性方程组的解法感兴趣的一个理由是, 能数值求解偏微分方程的重要性. 事实上, 许多目前通用的稀疏矩阵技术是由偏微分方程数值解的研究人员所提出的.

粗略地说, 解稀疏问题 $\boldsymbol{Ax}=\boldsymbol{b}$ 有两种方法. 一种方法是选直接法, 利用 $\boldsymbol{A}$ 的稀疏性作适当修改. 典型的修正策略涉及巧妙地利用数据存储结构和特殊的选主元技巧, 使得填充极小.

与直接法相对的是*迭代法*. 这类方法产生近似解序列 $\{\boldsymbol{x}^{(k)}\}$, 矩阵 $\boldsymbol{A}$ 只在矩阵与向量相乘时才用到. 迭代法的好坏主要集中体现在迭代序列 $\{\boldsymbol{x}^{(k)}\}$ 的收敛速度上. 在本节, 我们将给出几种基本的迭代方法, 讨论它们的实际实现, 并证明几个关于它们表现的具有代表性的定理.

### 10.1.1 Jacobi 迭代和 Gauss-Seidel 迭代

最简单的迭代法可能是 Jacobi *迭代*. 它对主对角元非零的矩阵才有意义. 在 $3\times 3$ 的问题 $\boldsymbol{Ax}=\boldsymbol{b}$ 中, 方程等价于:

$$\begin{aligned}
x_1 &= (b_1 - a_{12}x_2 - a_{13}x_3)/a_{11},\\
x_2 &= (b_2 - a_{21}x_1 - a_{23}x_3)/a_{22},\\
x_3 &= (b_3 - a_{31}x_1 - a_{32}x_2)/a_{33}.
\end{aligned}$$

假设 $\boldsymbol{x}^{(k)}$ 是 $\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$ 的一近似, 则产生新的近似解 $\boldsymbol{x}^{(k+1)}$ 的一种很自然的方法是计算:

$$\begin{aligned}
x_1^{(k+1)} &= (b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)})/a_{11},\\
x_2^{(k+1)} &= (b_2 - a_{21}x_1^{(k)} - a_{23}x_3^{(k)})/a_{22},\\
x_3^{(k+1)} &= (b_3 - a_{31}x_1^{(k)} - a_{32}x_2^{(k)})/a_{33}.
\end{aligned} \tag{10.1.1}$$

这就是 $n=3$ 时的 Jacobi 迭代. 对一般的 $n$, 迭代公式如下:

**for** $i=1:n$

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)\Bigg/ a_{ii} \tag{10.1.2}$$

**end**

注意, 在 Jacobi 迭代中, 计算 $x_i^{(k+1)}$ 时没有利用最新得到的信息. 例如, 即使 $x_1^{(k+1)}$ 已知, 计算 $x_2^{(k+1)}$ 也只是用到了 $x_1^{(k)}$. 如果我们修改 Jacobi 迭代, 使得每次都用准确值 $x_i$ 的最新估计, 则得如下算法:

**for** $i=1:n$

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)\Bigg/ a_{ii} \tag{10.1.3}$$

**end**

这就是所谓的 Gauss-Seidel 迭代.

上面两种迭代都可用矩阵 $\boldsymbol{L}, \boldsymbol{D}, \boldsymbol{U}$ 来简明地表达从 $\boldsymbol{x}^{(k)}$ 到 $\boldsymbol{x}^{(k+1)}$ 的转移. 定义

$$\boldsymbol{L} = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 \\ a_{21} & 0 & \cdots & & \vdots \\ a_{31} & a_{32} & \ddots & & 0 \\ \vdots & & & 0 & 0 \\ a_{n1} & a_{n2} & \cdots & a_{n,n-1} & 0 \end{bmatrix},$$

$$\boldsymbol{D} = \mathrm{diag}(a_{11}, \cdots, a_{nn}), \tag{10.1.4}$$

$$\boldsymbol{U} = \begin{bmatrix} 0 & a_{12} & \cdots & \cdots & a_{1n} \\ 0 & 0 & \cdots & & \vdots \\ 0 & 0 & \ddots & & a_{n-2,n} \\ \vdots & & & & a_{n-1,n} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

特别地, Jacobi 迭代具有形式 $\boldsymbol{M}_J\boldsymbol{x}^{(k+1)} = \boldsymbol{N}_J\boldsymbol{x}^{(k)} + \boldsymbol{b}$, 其中 $\boldsymbol{M}_J = \boldsymbol{D}, \boldsymbol{N}_J = -(\boldsymbol{L} + \boldsymbol{U})$. 另外, Gauss-Seidel 迭代可用 $\boldsymbol{M}_G\boldsymbol{x}^{(k+1)} = N_G\boldsymbol{x}^{(k)} + \boldsymbol{b}$ 来表达, 这时 $\boldsymbol{M}_G = \boldsymbol{L} + \boldsymbol{D}, \boldsymbol{N}_G = -\boldsymbol{U}$.

### 10.1.2　矩阵分裂和迭代收敛性

Jacobi 方法和 Gauss-Seidel 方法是一个大的迭代方法族:

$$\boldsymbol{M}\boldsymbol{x}^{(k+1)} = \boldsymbol{N}\boldsymbol{x}^{(k)} + \boldsymbol{b} \tag{10.1.5}$$

的典型代表, 这里 $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$ 是矩阵 $\boldsymbol{A}$ 的一个分裂. 为了使迭代 (10.1.5) 实用, 当 $\boldsymbol{M}$ 作为系数矩阵时线性方程组必须容易求解. 在 Jacobi 迭代和 Gauss-Seidel 迭代中, $\boldsymbol{M}$ 分别是对角矩阵和下三角形矩阵.

(10.1.5) 产生的迭代序列是否收敛到解 $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$ 取决于 $\boldsymbol{M}^{-1}\boldsymbol{N}$ 的特征值. 为此, 定义 $n \times n$ 矩阵 $\boldsymbol{G}$ 的谱半径为

$$\rho(\boldsymbol{G}) = \max\{|\lambda| : \lambda \in \lambda(\boldsymbol{G})\}.$$

$\rho(\boldsymbol{M}^{-1}\boldsymbol{N})$ 的大小对 (10.1.5) 的收敛性极为重要.

**定理 10.1.1**　设 $\boldsymbol{b} \in \mathbb{R}^n$ 和 $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N} \in \mathbb{R}^{n\times n}$ 非奇异, 如果 $\boldsymbol{M}$ 非奇异且 $\boldsymbol{M}^{-1}\boldsymbol{N}$ 的谱半径满足 $\rho(\boldsymbol{M}^{-1}\boldsymbol{N}) < 1$, 则对任何初始向量 $\boldsymbol{x}^{(0)}$, 由 $\boldsymbol{M}\boldsymbol{x}^{(k+1)} = \boldsymbol{N}\boldsymbol{x}^{(k)} + \boldsymbol{b}$ 产生的迭代序列 $\{\boldsymbol{x}^{(k)}\}$ 收敛到解 $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$.

**证明**　记 $\boldsymbol{e}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}$ 为第 $k$ 步迭代的误差. 由 $\boldsymbol{M}\boldsymbol{x} = \boldsymbol{N}\boldsymbol{x} + \boldsymbol{b}$ 可得 $\boldsymbol{M}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}) = \boldsymbol{N}(\boldsymbol{x}^{(k)} - \boldsymbol{x})$. 因此 $\boldsymbol{x}^{(k+1)}$ 中的误差为

$$\boldsymbol{e}^{(k+1)} = \boldsymbol{M}^{-1}\boldsymbol{N}\boldsymbol{e}^{(k)} = (\boldsymbol{M}^{-1}\boldsymbol{N})^{k+1}\boldsymbol{e}^{(0)}.$$

由引理 7.3.2 可知 $(\boldsymbol{M}^{-1}\boldsymbol{N})^k \to 0$ 当且仅当 $\rho(\boldsymbol{M}^{-1}\boldsymbol{N}) < 1$. □

对研究按下列思路所构造的算法, 这一结果是至关重要的.

- 把 $\boldsymbol{A}$ 分裂成 $\boldsymbol{A}=\boldsymbol{M}-\boldsymbol{N}$, 使得线性方程组 $\boldsymbol{Mz}=\boldsymbol{d}$ 容易求解;
- 鉴别出迭代矩阵 $\boldsymbol{G}=\boldsymbol{M}^{-1}\boldsymbol{N}$ 满足 $\rho(\boldsymbol{G})<1$ 的矩阵类.
- 给出关于 $\rho(\boldsymbol{G})$ 的一些进一步的结果, 分析误差 $\boldsymbol{e}^{(k)}$ 是如何趋近于零的.

例如, 考虑 Jacobi 迭代 $\boldsymbol{D}\boldsymbol{x}^{(k+1)}=-(\boldsymbol{L}+\boldsymbol{U})\boldsymbol{x}^{(k)}+\boldsymbol{b}$, 保证 $\rho(\boldsymbol{M}_J^{-1}\boldsymbol{N}_J)<1$ 的一个条件是严格对角占优. 实际上, 若 $\boldsymbol{A}$ 有此性质 (见 3.4.10 节), 则

$$\rho(\boldsymbol{M}_J^{-1}\boldsymbol{N}_J)\leqslant\|\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})\|_\infty=\max_{1\leqslant i\leqslant n}\sum_{\substack{j=1\\j\neq i}}^{n}\left|\frac{a_{ij}}{a_{ii}}\right|<1.$$

通常, 对角越占优, 收敛速度就越快, 但也有反例, 见习题 10.1.7.

为了证明 Gauss-Seidel 迭代对于对称正定矩阵是收敛的, 需要一个关于谱半径的更为复杂的结果.

**定理 10.1.2** *若 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 对称正定, 则对任何初始值 $\boldsymbol{x}^{(0)}$, Gauss-Seidel 迭代 (10.1.3) 收敛.*

**证明** 把 $\boldsymbol{A}$ 写成 $\boldsymbol{A}=\boldsymbol{L}+\boldsymbol{D}+\boldsymbol{L}^{\mathrm{T}}$, 其中 $\boldsymbol{D}=\mathrm{diag}(a_{ii})$, $\boldsymbol{L}$ 为严格下三角形矩阵. 根据定理 10.1.1, 只需证明矩阵 $\boldsymbol{G}=-(\boldsymbol{D}+\boldsymbol{L})^{-1}\boldsymbol{L}^{\mathrm{T}}$ 的特征值在单位圆内. 既然 $\boldsymbol{D}$ 是正定的, 我们有 $\boldsymbol{G}_1\equiv\boldsymbol{D}^{1/2}\boldsymbol{G}\boldsymbol{D}^{-1/2}=-(\boldsymbol{I}+\boldsymbol{L}_1)^{-1}\boldsymbol{L}_1^{\mathrm{T}}$, 其中 $\boldsymbol{L}_1=\boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2}$. 因为 $\boldsymbol{G}$ 与 $\boldsymbol{G}_1$ 的特征值相同, 我们只需证明 $\rho(\boldsymbol{G}_1)<1$. 若 $\boldsymbol{G}_1\boldsymbol{x}=\lambda\boldsymbol{x},\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}=1$, 则 $-\boldsymbol{L}_1^{\mathrm{T}}\boldsymbol{x}=\lambda(\boldsymbol{I}+\boldsymbol{L}_1)\boldsymbol{x}$, 因此 $-\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}_1^{\mathrm{T}}\boldsymbol{x}=\lambda(1+\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}_1\boldsymbol{x})$. 令 $\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}_1\boldsymbol{x}=a+\mathrm{i}b$, 则有

$$|\lambda|^2=\left|\frac{-a+b\mathrm{i}}{1+a+b\mathrm{i}}\right|^2=\frac{a^2+b^2}{1+2a+a^2+b^2}.$$

然而, 由于 $\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}=\boldsymbol{I}+\boldsymbol{L}_1+\boldsymbol{L}_1^{\mathrm{T}}$ 正定, 不难证明 $0<1+\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}_1\boldsymbol{x}+\boldsymbol{x}^{\mathrm{H}}\boldsymbol{L}_1^{\mathrm{T}}\boldsymbol{x}=1+2a$, 这意味着 $|\lambda|<1$. □

这一结果经常被引用, 因为很多椭圆型偏微分方程离散化以后得到的矩阵常是对称正定的. 在文献中, 出现了许多这类的结果.

### 10.1.3 Gauss-Seidel 迭代的实际实现

我们现在集中讨论几个关于 Gauss-Seidel 迭代的实用细节, 为了使算法实现起来简便, 把 Gauss-Seidel 迭代改写如下:

**for** $i=1:n$

$$x_i=\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j-\sum_{j=i+1}^{n}a_{ij}x_j\right)\bigg/a_{ii}$$

**end**

这一计算需要的 flop 数大约是 $\boldsymbol{A}$ 中非零元个数的两倍. 要得到工作量的更精确的结果是没有意义的, 因为算法的实际运行在很大程度上取决于相应问题中的矩阵结构.

为了强调这一点, 我们把 (10.1.3) 应用到 $NM \times NM$ 的块三对角系统:

$$\begin{bmatrix} \boldsymbol{T} & -\boldsymbol{I}_N & & \cdots & 0 \\ -\boldsymbol{I}_N & \boldsymbol{T} & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & -\boldsymbol{I}_N \\ 0 & \cdots & & -\boldsymbol{I}_N & \boldsymbol{T} \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ \vdots \\ g_M \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_M \end{bmatrix}, \tag{10.1.6}$$

其中 $T = \begin{bmatrix} 4 & -1 & & \cdots & 0 \\ -1 & 4 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & -1 \\ 0 & \cdots & & -1 & 4 \end{bmatrix}, g_j = \begin{bmatrix} G(1,j) \\ G(2,j) \\ \vdots \\ \vdots \\ G(N,j) \end{bmatrix}, f_j = \begin{bmatrix} F(1,j) \\ F(2,j) \\ \vdots \\ \vdots \\ F(N,j) \end{bmatrix}.$

当 Poisson 方程在一长方形上离散化以后产生的就是这种问题. 容易证明矩阵 $\boldsymbol{A}$ 是正定的.

当 $i \in \{0, N+1\}$ 或 $j \in \{0, M+1\}$ 时约定 $G(i,j) = 0$, 可以把 Gauss-Seidel 迭代写成如下形式:

**for** $j = 1 : M$

　　**for** $i = 1 : N$

$$\begin{aligned} G(i,j) &= (F(i,j) + G(i-1,j) + G(i+1,j) \\ &\quad + G(i,j-1) + G(i,j+1))/4 \end{aligned}$$

　　**end**

**end**

注意, 在这个问题中, 对矩阵 $\boldsymbol{A}$ 没有存储要求.

### 10.1.4　逐次超松弛迭代

Gauss-Seide 迭代由于其简单而引人注目. 不幸的是, 当 $\boldsymbol{M}_G^{-1}\boldsymbol{N}_G$ 的谱半径接近于 1 时, 收敛将会很慢, 因为误差 $e^{(k)}$ 以 $\rho(\boldsymbol{M}_G^{-1}\boldsymbol{N}_G)^k$ 的速度趋近于零. 为了改进这一点, 设 $w \in \mathbb{R}$, 把 Gauss-Seidel 迭代作如下修正:

**for** $i = 1 : n$

$$x_i^{(k+1)} = w\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right) \bigg/ a_{ii} + (1-w)x_i^{(k)} \tag{10.1.7}$$

**end**

这就是*逐次超松弛*(SOR) 方法. 利用 (10.1.4) 中的记号, 可以看出 SOR 迭代由下述方程给出:

$$M_w x^{(k+1)} = N_w x^{(k)} + wb, \tag{10.1.8}$$

其中 $M_w = D + wL, N_w = (1-w)D - wU$. 对少数几个有结构的 (但很重要) 问题, 如 (10.1.6), 使得 $\rho(M_w^{-1}N_w)$ 极小化的松弛参数$w$ 的值是已知的. 此外, 可产生 $\rho(M_1^{-1}N_1) = \rho(M_G^{-1}N_G)$ 的一个重要约化. 但是, 在更复杂的问题中, 要决定一个合适的 $w$, 需要进行非常复杂的特征值分析.

### 10.1.5 Chebyshev 半迭代方法

另一种加速迭代收敛速度的方法是利用 Chebyshev(切比雪夫) 多项式. 假设 $x^{(1)}, x^{(2)}, \cdots, x^{(k)}$ 是由迭代 $Mx^{(j+1)} = Nx^{(j)} + b$ 产生, 我们希望确定系数 $\nu_j(k), j = 0:k$, 使得

$$y^{(k)} = \sum_{j=0}^{k} \nu_j(k) x^{(j)} \tag{10.1.9}$$

比 $x^{(k)}$ 更好. 若 $x^{(0)} = \cdots = x^{(k)} = x$. 很自然要求 $y^{(k)} = x$. 因此要求

$$\sum_{j=0}^{k} \nu_j(k) = 1. \tag{10.1.10}$$

在此限制下, 我们考虑如何选取 $\nu_j(k)$ 使得 $y^{(k)}$ 的误差极小化.

回忆定理 10.1.1 的证明, $x^{(k)} - x = (M^{-1}N)^k e^{(0)}$, 其中 $e^{(0)} = x^{(0)} - x$, 可知

$$y^{(k)} - x = \sum_{j=0}^{k} \nu_j(k)(x^{(j)} - x) = \sum_{j=0}^{k} \nu_j(k)(M^{-1}N)^j e^{(0)}.$$

两边取范数 $\|\cdot\|_2$ 得

$$\| y^{(k)} - x \|_2 \leqslant \| p_k(G) \|_2 \| e^{(0)} \|_2, \tag{10.1.11}$$

其中 $G = M^{-1}N$,

$$p_k(z) = \sum_{j=0}^{k} \nu_j(k) z^j.$$

由条件 (10.1.10) 知 $p_k(1) = 1$.

在此基础上假设 $G$ 是对称的且特征值 $\lambda_i$ 满足:

$$-1 < \alpha \leqslant \lambda_n \leqslant \cdots \leqslant \lambda_1 \leqslant \beta < 1.$$

从而有

$$\| p_k(G) \|_2 = \max_{\lambda_i \in \lambda(A)} |p_k(\lambda_i)| \leqslant \max_{\alpha \leqslant \lambda \leqslant \beta} |p_k(\lambda)|.$$

因此, 为了使得 $p_k(G)$ 的范数很小, 我们需要寻找一多项式 $p_k(z)$ 使其在区间 $[\alpha, \beta]$ 上尽可能小且满足 $p_k(1) = 1$.

考虑由递推公式 $c_j(z) = 2zc_{j-1}(z) - c_{j-2}(z), c_0(z) = 1, c_1(z) = z$ 产生的 Chebyshev 多项式 $c_j(z)$. 它们在区间 $[-1, 1]$ 上满足 $|c_j(z)| \leqslant 1$, 但在区间 $[-1, 1]$ 之外, 上升很快. 因此, 多项式 $p_k(z)$ 取为

$$p_k(z) = c_k\left(-1+2\frac{z-\alpha}{\beta-\alpha}\right)\Big/ c_k(\mu),$$

其中 $\mu=-1+2\dfrac{1-\alpha}{\beta-\alpha}=1+2\dfrac{1-\beta}{\beta-\alpha}$, 它满足 $p_k(1)=1$ 且在 $[\alpha,\beta]$ 上很小. 从 $p_k(z)$ 的定义及等式 (10.1.11) 可知

$$\| \boldsymbol{y}^{(k)} - \boldsymbol{x} \|_2 \leqslant \frac{\| \boldsymbol{x}-\boldsymbol{x}^{(0)} \|_2}{|c_k(\mu)|}.$$

因此, $\mu$ 越大, 收敛的加速也越快.

为使上述讨论成为一个实用的加速算法, 需要比 (10.1.9) 更有效的方法来计算 $\boldsymbol{y}^{(k)}$. 我们已假设 $n$ 是很大的, 因此, 对很大的 $k$, $\boldsymbol{x}^{(0)},\boldsymbol{x}^{(1)},\cdots,\boldsymbol{x}^{(k)}$ 的重复利用是不方便的, 有时甚至是不可能的.

幸运的是, 利用 Chebyshev 多项式中的三项递推公式, 可导出 $y^{(k)}$ 的三项递推计算公式. 特别地, 可以证明: 若

$$w_{k+1} = 2\frac{2-\beta-\alpha}{\beta-\alpha}\frac{c_k(\mu)}{c_{k+1}(\mu)},$$

则

$$\begin{aligned} &\boldsymbol{y}^{(k+1)} = w_{k+1}(\boldsymbol{y}^{(k)}-\boldsymbol{y}^{(k-1)}+\gamma \boldsymbol{z}^{(k)})+\boldsymbol{y}^{(k-1)},\\ &\boldsymbol{M}\boldsymbol{z}^{(k)} = \boldsymbol{b}-\boldsymbol{A}\boldsymbol{y}^{(k)},\\ &\gamma = 2/(2-\alpha-\beta), \end{aligned} \tag{10.1.12}$$

其中 $\boldsymbol{y}^{(0)}=\boldsymbol{x}^{(0)},\boldsymbol{y}^{(1)}=\boldsymbol{x}^{(1)}$. 我们称这一技巧为相应于 $\boldsymbol{M}\boldsymbol{y}^{(k+1)}=\boldsymbol{N}\boldsymbol{y}^{(k)}+\boldsymbol{b}$ 的 Chebyshev 半迭代方法. 为了使加速有效, 需要好的下界 $\alpha$ 与上界 $\beta$. 和 SOR 方法一样, 除了少数几个有结构的问题之外, 这些参数很难确定.

Varga (1962, 第 5 章), Golub and Varga (1961) 深入地分析了 Chebyshev 半迭代方法.

### 10.1.6 对称 SOR 方法

在导出 Chebyshev 加速技巧时, 我们假设了迭代矩阵 $\boldsymbol{G}=\boldsymbol{M}^{-1}\boldsymbol{N}$ 对称. 因此, 上面简单的分析不能应用于非对称 SOR 迭代矩阵 $\boldsymbol{M}_w^{-1}\boldsymbol{N}_w$. 然而, 可以将 SOR 方法对称化, 使之可利用 Chebyshev 加速技巧. 其思想是将 SOR 方法与向后SOR 方法相结合:

**for** $i=n:-1:1$

$$x_i^{(k+1)} = w\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j^{(k+1)}-\sum_{j=i+1}^{n}a_{ij}x_j^{(k)}\right)\Big/ a_{ii}+(1-w)x_i^{(k)} \tag{10.1.13}$$

**end**

在 (10.1.7) 中, 把未知数的顺序颠倒过来, 就得到上面的迭代. 向后 SOR 方法可用 (10.1.4) 中的矩阵记号来描述. 特别地, 有

$$\widetilde{\boldsymbol{M}}_w x^{(k+1)} = \widetilde{\boldsymbol{N}}_w x^{(k)} + wb,$$

其中

$$\widetilde{\boldsymbol{M}}_w = \boldsymbol{D} + w\boldsymbol{U}, \quad \widetilde{\boldsymbol{N}}_w = (1-w)\boldsymbol{D} - w\boldsymbol{L}. \tag{10.1.14}$$

如果 $\boldsymbol{A}$ 对称 $(\boldsymbol{U} = \boldsymbol{L}^{\mathrm{T}})$, 则 $\widetilde{\boldsymbol{M}}_w = \boldsymbol{M}_w^{\mathrm{T}}, \widetilde{\boldsymbol{N}}_w = \boldsymbol{N}_w^{\mathrm{T}}$ 且有如下迭代:

$$\begin{aligned} \boldsymbol{M}_w x^{(k+1/2)} &= \boldsymbol{N}_w x^{(k)} + wb, \\ \boldsymbol{M}_w^{\mathrm{T}} x^{(k+1)} &= \boldsymbol{N}_w^{\mathrm{T}} x^{(k+1/2)} + wb. \end{aligned} \tag{10.1.15}$$

很显然, $\boldsymbol{G} = \boldsymbol{M}_w^{-\mathrm{T}}\boldsymbol{N}_w^{\mathrm{T}}\boldsymbol{M}_w^{-1}\boldsymbol{N}_w$ 是这一方法的迭代矩阵. 从 $\boldsymbol{M}_w$ 和 $\boldsymbol{N}_w$ 的定义可得

$$\boldsymbol{G} = \boldsymbol{M}^{-1}\boldsymbol{N} \equiv (\boldsymbol{M}_w\boldsymbol{D}^{-1}\boldsymbol{M}_w^{\mathrm{T}})^{-1}(\boldsymbol{N}_w^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{N}_w). \tag{10.1.16}$$

如果 $\boldsymbol{D}$ 的对角元都是正的, 且$\boldsymbol{K}\boldsymbol{K}^{\mathrm{T}}$为$\boldsymbol{N}_w^{\mathrm{T}}\boldsymbol{D}^{-1}\boldsymbol{N}_w$的 Cholesky 分解, 则$\boldsymbol{K}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{K}^{-\mathrm{T}} = \boldsymbol{K}^{\mathrm{T}}(\boldsymbol{M}_w\boldsymbol{D}^{-1}\boldsymbol{M}_w^{\mathrm{T}})^{-1}\boldsymbol{K}$. 因此, $\boldsymbol{G}$ 与一对称矩阵相似, 特征值为实的.

迭代 (10.1.15) 称为**对称超松弛迭代**(SSOR) 方法. 它常常和 Chebyshev 半迭代加速技巧在一起用.

## 习 题

**10.1.1** 证明: Jacobi 迭代可写成 $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \boldsymbol{H}\boldsymbol{r}^{(k)}$ 的形式, 其中 $\boldsymbol{r}^{(k)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(k)}$. 重新写出 Gauss-Seidel 迭代的形式.

**10.1.2** 证明: 若矩阵 $\boldsymbol{A}$ 严格对角占优, 则 Gauss-Seidel 迭代收敛.

**10.1.3** 证明: 对 2×2 对称正定矩阵, Jacobi 迭代收敛.

**10.1.4** 证明: 若 $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$ 是奇异的, 则既使 $\boldsymbol{M}$ 非奇异, 也不可能有 $\rho(\boldsymbol{M}^{-1}\boldsymbol{N}) < 1$ 成立.

**10.1.5** 证明 (10.1.16).

**10.1.6** 证明定理 10.1.1 的逆命题. 也就是说, 若迭代 $\boldsymbol{M}x^{(k+1)} = \boldsymbol{N}x^{(k)} + b$ 总是收敛的, 则 $\rho(\boldsymbol{M}^{-1}\boldsymbol{N}) < 1$.

**10.1.7** (由 R. S. Varga 提供) 假设

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & -1/2 \\ -1/2 & 1 \end{bmatrix}, \quad \boldsymbol{A}_2 = \begin{bmatrix} 1 & -3/4 \\ -1/12 & 1 \end{bmatrix},$$

$\boldsymbol{J}_1$ 和 $\boldsymbol{J}_2$ 分别是相应的 Jacobi 迭代矩阵, 证明 $\rho(\boldsymbol{J}_1) > \rho(\boldsymbol{J}_2)$, 并反驳如下结论: 对角元占优程度越大, Jacobi 迭代收敛的速度越快.

**10.1.8** Chebyshev 算法由下面参数确定:

$$w_{k+1} = \frac{2c_k(1/\rho)}{\rho c_{k+1}(1/\rho)},$$

这里 $c_k(\lambda) = \cosh[k \cosh^{-1}(\lambda)], \lambda > 1$.

(a) 证明: 只要 $0 < \rho < 1$, 则对 $k > 1$ 有 $1 < w_k < 2$.

(b) 证明: $w_{k+1} < w_k$.

(c) 确定极限 $\lim\limits_{k\to\infty} w_k$.

**10.1.9** 考虑 2×2 矩阵 $\boldsymbol{A} = \begin{bmatrix} 1 & \rho \\ -\rho & 1 \end{bmatrix}$.

(a) 在什么情况下, Gauss-Seidel 迭代收敛?

(b) 对于什么取值范围的参数 $w$, SOR 方法收敛? 这一参数的最优选取是何值?

(C) 对矩阵 $\boldsymbol{A}=\begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{S} \\ -\boldsymbol{S}^{\mathrm{T}} & \boldsymbol{I}_n \end{bmatrix}$, 重复 (a)(b) 的问题 (提示: 利用 $\boldsymbol{S}$ 的奇异值分解).

**10.1.10** 我们要求方程 $\boldsymbol{Au}=\boldsymbol{f}$ 的解, 其中 $\boldsymbol{A}\neq\boldsymbol{A}^{\mathrm{T}}$, 一个典型的问题是考虑下列方程的有限差分逼近:

$$\begin{cases} -u''+\sigma u'=0, & 0<x<1, \\ u_{(0)}=10, & u_{(1)}=10\mathrm{e}^{\sigma}. \end{cases}$$

得到的差分方程如下:

$$-u_{i-1}+2u_i-u_{i+1}+R(u_{i+1}-u_{i-1})=0, \quad i=1:n,$$

其中 $R=\sigma h/2, u_0=10$ 且 $u_{n+1}=10\mathrm{e}^{\sigma}$, 数 $R$ 应当小于 1, 迭代 $\boldsymbol{Mu}^{(k+1)}=\boldsymbol{Nu}^{(k)}+\boldsymbol{f}$ 的收敛速度是多少? 其中 $\boldsymbol{M}=(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})/2, \boldsymbol{N}=(\boldsymbol{A}^{\mathrm{T}}-\boldsymbol{A})/2$.

**10.1.11** 考虑迭代: $y^{(k+1)}=w(\boldsymbol{By}^{(k)}+\boldsymbol{d}-\boldsymbol{y}^{(k-1)})+\boldsymbol{y}^{(k-1)}$, 其中 $\boldsymbol{B}$ 的 Schur 分解为 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{BQ}=\mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_n), \lambda_1\geqslant\cdots\geqslant\lambda_n$, 假设 $\boldsymbol{x}=\boldsymbol{Bx}+\boldsymbol{d}$.

(a) 导出一个关于 $\boldsymbol{e}^{(k)}=\boldsymbol{y}^{(k)}-\boldsymbol{x}$ 的方程.

(b) 假定 $\boldsymbol{y}^{(1)}=\boldsymbol{By}^{(0)}+\boldsymbol{d}$, 证明 $\boldsymbol{e}^{(k)}=\boldsymbol{p}_k(\boldsymbol{B})\boldsymbol{e}^{(0)}$, 其中当 $k$ 为偶数时, $p_k$ 为一偶次多项式, $k$ 为奇数为, $p_k$ 为一奇次多项式.

(c) 令 $\boldsymbol{f}^{(k)}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{e}^{(k)}$, 导出一个关于 $\boldsymbol{f}_j^{(k)}(j=1:n)$ 的差分方程, 试确定 $\boldsymbol{f}_j^{(0)}$ 和 $\boldsymbol{f}_j^{(1)}$ 的准确解.

(d) 说明如何选取最优值 $w$.

## 本节注释与参考文献

正如我们所指出的, Young(1971) 对 SOR 方法有最全面的处理. "SOR 理论" 的目标是指导用户选择松弛参数 $w$. 在这种情况下, 方程和未知数的排列是很重要的. 见:

M. J. M. Bernal and J. H. Verner (1968). "On Generalizing of the Theory of Consistent Orderings for Successive Over-Relaxation Methods," *Numer. Math.* 12, 215–222.

D. M. Young(1970). "Convergence Properties of the Symmetric and Unsymmetric Over-Relaxation Methods," *Math. Comp.* 24, 793–807.

D. M. Young(1972). "Generalization of Property A and Consistent Ordering," *SIAM J. Num. Anal.* 9, 454–463.

R. A. Nicolaides(1974). "On a Geometrical Aspect of SOR and the Theory of Consistent Ordering for Positive Definite Matrices," *Numer. Math.* 12, 99–104.

L. Adams and H. Jordan(1986). "Is SOR Color-Blind?" *SIAM J. Sci. Stat. Comp.* 7, 490-506.

M. Eiermann and R. S. Varga(1993). "Is the Optimal $w$ Best for the SOR Iteration Method," *Lin. Alg. and Its Applic.* 182, 257–277.

有关 Chebyshev 半迭代法的分析可参见:

G. H. Golub and R. S. Varga(1961). "Chebychev Semi-Iterative Methods, Successive Over-Relaxation Iterative Methods, and Second-Order Richardson Iterative Methods, Parts I and II," *Numer. Math.* 3, 147–156, 157–168.

这一工作需要一前提, 即迭代矩阵要有实特征值. 当不是这种情况时, 讨论可见:

T. A. Manteuffel (1977). "The Tchebychev Iteration for Nonsymmetric Linear Systems," *Numer. Math.* 28, 307–327.

M.Eiermann and W. Niethammer(1983). "On the Construction of Semi-iterative Methods," *SIAM J. Numer. Anal.* 20, 1153–1160.

W.Niethammer and R. S. Varga(1983). "The Analysis of k-step Iterative Methods for Linear Systems from Summability Theory," *Numer. Math.* 41, 177–206.

G. H. Golub and M. Overton(1988). "The Convergence of Inexact Chebychev and Richardson Iterative Methods for Solving Linear Systems," *Numer.Math.* 53, 571–594.

D. Calvetti, G. H. Golub, and L. Reichel(1994). "An Adaptive Chebyshev Iterative Method for Nonsymmetric Linear Systems Based on Modified Moments," *Numer. Math.* 67, 21–40.

其他的非对称方法包括:

M. Eiermann, W. Niethammer, and R. S. Varga(1992), "Acceleration of Relaxation Methods for Non-Hermitian Linear Systems," *SIAM J. Matrix Anal. Appl.* 13, 979–991.

H. Elman and G. H. Golub(1990). "Iterative Methods for Cyclically Reduced Non-Self-Adjoint Linear Systems I," *Math. Comp.* 54, 671–700.

H. Elman and G. H. Golub(1990). "Iterative Methods for Cyclically Reduced Non-Self-Adjoint Linear Systems II," *Math. Comp.* 56, 215–242.

R. Bramley and A. Sameh(1992). "Row projection Methods for Large Nonsymmetric linear Systems," *SIAM J. Sci. Statist. Comput.* 13, 168–193.

既然所有相关的特征值都是实的, 所以有时有可能把一迭代法对称化, 从而简化加速过程. 这就是 SSOR 方法的情形, 讨论见:

J. W. Sheldon(1955). "On the Numerical Solution of Elliptic Difference Equations," *Math. Tables Aids Comp.* 9, 101–112.

经典迭代法的并行实现也已引起人们注意, 见:

D. J. Evans(1984). "Parallel SOR Iterative Methods," *Parallel Computing* 1, 3–18.

N. Patel and H. Jordan(1984). "A Parallelized Point Rowwise Successive Over-Relaxation Method on a Multiprocessor," *Parallel Computing* 1, 207–222.

R. J.Plemmons(1986). "A Parallel Block Iterative Scheme Applied to Computations in Structural Analysis," *SLAM J. Alg. and Disc. Methods* 7, 337–347.

C. Kamath and A. Sameh(1989). "A Projection Method for Solving Nonsymmetric Linear Systems on Multiprocessors," *Parallel Computing* 9, 291–312.

我们知道, 用直接法解 $Ax = b$ 时, 条件数 $\kappa(A)$ 是一个重要的问题. 然而, 方程的条件数对迭代法也有影响, 见:

M. Arioli and F. Romani(1985)."Relations Between Condition Numbers and the Convergence of the Jacobi Method for Real positive Definite Matrices," *Numer. Math.* 46, 31–42.

M. Arioli, I. S. Duff, and D. Ruiz(1992). "Stopping Criteria for Iterative Solvers," *SIAM J. Matrix Anal. Appl.* 13, 138–144.

有关奇异方程组迭代法的讨论见：

A. Dax(1990)."The Convergence of Linear Stationary Iterative Processes for Solving Singular Unstructured Systems of Linear Equations," *SLAN Review* 32，611–635.

最后, 本节的方法之截断误差影响在下文中讨论.

H. Wozniakowski(1978). "Roundoff-Error Analysis of Iterations for Large Linear Systems," *Numer. Math.* 30, 301–314.

P. A. Knight(1993)."Error Analysis of Stationary Iteration and Associated Problems," Ph. D. thesis, Department of Mathematics, University of Manchester, England.

## 10.2 共轭梯度法

SOR 方法、Chebyshev 半迭代法以及相关方法的一个困难是它们依赖于参数, 而适当地选取这些参数有时是很难的. 例如, 在 Chebyshev 加速技巧中要估计迭代矩阵 $M^{-1}N$ 的最大、最小特征值. 除非该矩阵有充分的结构, 否则要做到这一点是从解析上不可能或在计算上太昂贵.

在本节, 对于对称正定问题 $Ax = b$, 我们给出避免上述困难的一个方法, 即著名的 Hestenes-Stiefel 共轭梯度法. 在 9.3.1 节中, 我们已从 Lanczos 算法中导出过这一方法. 现在, 从另一角度来导出此方法, 此推导也为 10.3 节和 10.4 节中的几种重要推广作准备.

### 10.2.1 最速下降法

推导的出发点是考虑如何极小化下列函数：

$$\phi(x) = \frac{1}{2}x^{\mathrm{T}}Ax - x^{\mathrm{T}}b,$$

其中 $b \in \mathbb{R}^n, A \in \mathbb{R}^{n\times n}$, 假设 $A$ 对称正定. $\phi(x)$ 的极小值是 $\dfrac{-b^{\mathrm{T}}A^{-1}b}{2}$, 令$x = A^{-1}b$ 即可得到. 因此, 当 $A$ 对称正定时, 极小化 $\phi(x)$ 和解方程 $Ax = b$ 是等价的.

极小化 $\phi(x)$ 的最简单的策略之一是**最速下降法**. 对当前点 $x_c, \phi(x)$ 在负梯度方向 $-\nabla\phi(x_c) = b - Ax_c$ 上下降最快, 我们称 $r_c = b - Ax_c$ 为 $x_c$ 的残量. 如果残量非零, 则存一正数 $\alpha$, 使得 $\phi(x_c + \alpha r_c) < \phi(x_c)$. 在最速下降法 (用精确线搜索) 中, 令

$$\alpha = r_c^{\mathrm{T}}r_c/r_c^{\mathrm{T}}Ar_c,$$

则函数

$$\phi(\boldsymbol{x}_c+\alpha\boldsymbol{r}_c)=\phi(\boldsymbol{x}_c)-\alpha\boldsymbol{r}_c^{\mathrm{T}}\boldsymbol{r}_c+\frac{1}{2}\alpha^2\boldsymbol{r}_c^{\mathrm{T}}\boldsymbol{A}\boldsymbol{r}_c,$$

达到极小. 我们给出如下算法:

$$\begin{aligned}
&x_0 = \text{初值}\\
&r_0 = b-\boldsymbol{A}x_0\\
&k=0\\
&\textbf{while } r_k\neq 0\\
&\qquad k=k+1\\
&\qquad \alpha_k = r_{k-1}^{\mathrm{T}}r_{k-1}/r_{k-1}^{\mathrm{T}}Ar_{k-1}\\
&\qquad x_k = x_{k-1}+\alpha_k r_{k-1}\\
&\qquad r_k = b-Ax_k\\
&\textbf{end}
\end{aligned}\tag{10.2.1}$$

可以证明

$$\left(\phi(\boldsymbol{x}_k)+\frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b}\right)\leqslant\left(1-\frac{1}{\kappa_2(\boldsymbol{A})}\right)\left(\phi(\boldsymbol{x}_{k-1})+\frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b}\right),\tag{10.2.2}$$

这说明算法是全局收敛的. 不幸的是, 当条件数 $\kappa_2(\boldsymbol{A})=\lambda_1(A)/\lambda_n(\boldsymbol{A})$ 很大时, 收敛的速度是极慢的. 在几何上, 这意味着 $\phi(\boldsymbol{x})$ 的等高线是一压的很偏的超椭球. 找最小点, 也就是找一个两边陡峭中间相对平坦的山谷之最低点. 在最速下降法中, 翻来覆去地穿越山谷, 而不是顺着谷而下. 换句话说, 迭代中梯度方向的变化不是足够大.

### 10.2.2 一般的搜索方向

为了避免最速下降法中的缺陷, 我们考虑沿着一组方向 $\{\boldsymbol{p}_1,\boldsymbol{p}_2,\cdots\}$ 逐步极小化 $\phi$, 它们不必是残量方向 $\{\boldsymbol{r}_0,\boldsymbol{r}_1,\cdots\}$. 易证, 当 $\alpha=\alpha_k=\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{r}_{k-1}/\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k$ 时, $\phi(\boldsymbol{x}_{k-1}+\alpha\boldsymbol{p}_k)$ 达到极小. 若这样选取 $\alpha$, 可证

$$\phi(\boldsymbol{x}_{k-1}+\alpha_k\boldsymbol{p}_k)=\phi(\boldsymbol{x}_{k-1})-\frac{1}{2}\frac{\left(\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{r}_{k-1}\right)^2}{\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k}.\tag{10.2.3}$$

为了保证 $\phi(\boldsymbol{x})$ 的值能减小, $\boldsymbol{p}_k$ 与 $\boldsymbol{r}_{k-1}$ 不能正交. 这就导出了算法的框架:

$$\begin{aligned}
&x_0 = \text{初值}\\
&r_0 = b-Ax_0\\
&k=0\\
&\textbf{while } r_k\neq 0\\
&\qquad k=k+1\\
&\qquad \text{选一方向 } p_k, \text{使得 } p_k^{\mathrm{T}}r_{k-1}\neq 0\\
&\qquad \alpha_k = p_k^{\mathrm{T}}r_{k-1}/p_k^{\mathrm{T}}Ap_k\\
&\qquad x_k = x_{k-1}+\alpha_k p_k
\end{aligned}\tag{10.2.4}$$

$r_k = b - Ax_k$

**end**

注意, $\boldsymbol{x}_k \in \boldsymbol{x}_0 + \text{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\} \equiv \{\boldsymbol{x}_0 + \gamma_1 \boldsymbol{p}_1 + \cdots + \gamma_k \boldsymbol{p}_k : \gamma_i \in \mathbb{R}\}$. 我们的目标是选择一列既保证收敛性, 又能避免最速下降法缺点的搜索方向.

### 10.2.3　A 共轭搜索方向

如果搜索方向是线性无关的, 且 $\boldsymbol{x}_k$ 为下列问题的解

$$\min_{\boldsymbol{x} \in \boldsymbol{x}_0 + \text{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}} \phi(\boldsymbol{x}) \tag{10.2.5}$$

$k = 1, 2, \cdots$, 则能保证最多 $n$ 步就收敛到真解. 这是因为 $\boldsymbol{x}_n$ 使 $\phi(\boldsymbol{x})$ 在 $\mathbb{R}^n$ 上极小, 从而 $\boldsymbol{A}\boldsymbol{x}_n = \boldsymbol{b}$.

然而, 为了使这成为一种可行的途径, 搜索方向必须具备这样的性质: 能够很容易从 $\boldsymbol{x}_{k-1}$ 计算出 $\boldsymbol{x}_k$. 让我们考察一下, 在上述条件下怎样决定 $p_k$. 若

$$\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y} + \alpha \boldsymbol{p}_k,$$

其中 $\boldsymbol{P}_{k-1} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k-1}], \boldsymbol{y} \in \mathbb{R}^{k-1}, \alpha \in \mathbb{R}$, 则

$$\phi(\boldsymbol{x}_k) = \phi(\boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y}) + \alpha \boldsymbol{y}^{\mathrm{T}} \boldsymbol{P}_{k-1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k + \frac{\alpha^2}{2} \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k - \alpha \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_0.$$

如果 $\boldsymbol{p}_k \in \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\}^{\perp}$, 则交叉项 $\alpha \boldsymbol{y}^{\mathrm{T}} \boldsymbol{P}_{k-1}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k$ 为零, 且寻找极小化点 $\boldsymbol{x}_k$ 化为两个独立的极小化问题: 一个是寻找 $\boldsymbol{y}$, 一个是寻找 $\alpha$, 即

$$\begin{aligned}
\min_{\boldsymbol{x}_k \in \boldsymbol{x}_0 + \text{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\}} \phi(\boldsymbol{x}_k) &= \min_{\boldsymbol{y}, \alpha} \phi(\boldsymbol{x}_0 + P_{k-1}\boldsymbol{y} + \alpha \boldsymbol{p}_k) \\
&= \min_{\boldsymbol{y}, \alpha} \left( \phi(\boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y}) + \frac{\alpha^2}{2} \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k - \alpha \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_0 \right) \\
&= \min_{\boldsymbol{y}} \phi(\boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y}) + \min_{\alpha} \left( \frac{\alpha^2}{2} \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k - \alpha \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_0 \right).
\end{aligned}$$

注意, 若 $\boldsymbol{y}_{k-1}$ 是第一个极小化问题的解, 则 $\boldsymbol{x}_{k-1} = \boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y}_{k-1}$ 在 $\boldsymbol{x}_0 + \text{span}$ $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{k-1}\}$ 上极小化 $\phi(\boldsymbol{x})$. 对 $\alpha$ 的极小化问题之解是 $\alpha_k = \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_0 / \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{A} \boldsymbol{p}_k$. 由 $\boldsymbol{A}$ 共轭条件有

$$\begin{aligned}
\boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_{k-1} &= \boldsymbol{p}_k^{\mathrm{T}} (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{k-1}) \\
&= \boldsymbol{p}_k^{\mathrm{T}} (\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_0 + \boldsymbol{P}_{k-1}\boldsymbol{y}_{k-1})) = \boldsymbol{p}_k^{\mathrm{T}} \boldsymbol{r}_0.
\end{aligned}$$

利用这些结果可知 $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k \boldsymbol{p}_k$, 且得到了 (10.2.4) 的一个例子:

$x_0 =$ 初值

$k = 0$

$r_0 = b - Ax_0$

**while** $r_k \neq 0$

　　$k = k + 1$

　　选 $p_k \in \text{span}\{Ap_1, \cdots, Ap_{k-1}\}^{\perp}$, 且 $p_k^{\mathrm{T}} r_{k-1} \neq 0$　　(10.2.6)

　　$\alpha_k = p_k^{\mathrm{T}} r_{k-1} / p_k^{\mathrm{T}} A p_k$

$x_k = x_{k-1} + \alpha_k p_k$

$r_k = b - Ax_k$

**end**

下面的引理表明找到满足以上性质的搜索方向是可能的.

**引理 10.2.1** 若 $\boldsymbol{r}_{k-1} \neq \boldsymbol{0}$, 则存在 $\boldsymbol{p}_k \in \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\}^{\perp}$ 且 $\boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{r}_{k-1} \neq 0$.

**证明** 当 $k=1$ 时, 令 $\boldsymbol{p}_1 = \boldsymbol{r}_0$. 当 $k>1$ 时, 由 $\boldsymbol{r}_{k-1} \neq \boldsymbol{0}$ 可知

$$\boldsymbol{A}^{-1}\boldsymbol{b} \notin \boldsymbol{x}_0 + \text{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k-1}\}$$

$$\Rightarrow \boldsymbol{b} \notin \boldsymbol{A}\boldsymbol{x}_0 + \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\} \Rightarrow \boldsymbol{r}_0 \notin \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\}.$$

因此存在 $\boldsymbol{p} \in \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\}^{\perp}$ 使得 $\boldsymbol{p}^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$. 但 $\boldsymbol{x}_{k-1} \in \boldsymbol{x}_0 + \text{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k-1}\}$, 所以 $\boldsymbol{r}_{k-1} \in \boldsymbol{r}_0 + \text{span}\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\}$. 从而可得 $\boldsymbol{p}^{\mathrm{T}}\boldsymbol{r}_{k-1} = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{r}_0 \neq 0$. □

(10.2.6) 中的搜索方向称为 A 共轭的, 因为对所有的 $i \neq j$, 有 $\boldsymbol{p}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_j = 0$. 如果 $\boldsymbol{P}_k = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k]$ 是由这些向量组成的矩阵, 则

$$\boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_k = \text{diag}(\boldsymbol{p}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k)$$

非奇异, 这是因为 $\boldsymbol{A}$ 是正定的且搜索方向非零. 由此可见, $\boldsymbol{P}_k$ 列满秩. 这保证了最多 $n$ 步收敛到真解, 因为 $\boldsymbol{x}_n$(如果迭代到这一步的话) 在 $\text{ran}(\boldsymbol{P}_n) = \mathbb{R}^n$ 上极小化 $\phi(\boldsymbol{x})$.

### 10.2.4 选择最佳搜索方向

一种结合最速下降法和 A 共轭搜索方向法优点的方法是在 (10.2.5) 中选取离 $\boldsymbol{r}_{k-1}$ 最近且与 $\boldsymbol{p}_1, \cdots, \boldsymbol{p}_{k-1}$ A 共轭的向量 $\boldsymbol{p}_k$. 这就得到了共轭梯度法的“版本 0”:

$x_0 =$ 初值

$k = 0$

$r_0 = b - Ax_0$

**while** $r_k \neq 0$

  $k = k + 1$

  **if** $k = 1$     (10.2.7)

    $p_1 = r_0$

  **else**

    选取 $p_k$, 使得它在所有向量 $p \in \text{span}\{Ap_1, \cdots, Ap_{k-1}\}^{\perp}$ 上极小化 $\| p - r_{k-1} \|_2$

  **end**

  $\alpha_k = p_k^{\mathrm{T}} r_{k-1} / p_k^{\mathrm{T}} A p_k$

  $x_k = x_{k-1} + \alpha_k p_k$

  $r_k = b - Ax_k$

**end**

$x = x_k$

为了使该算法成为稀疏问题 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的有效解法, 我们需要计算 $\boldsymbol{p}_k$ 的有效方法. 为了得到最终的迭代公式, 还需大量的分析. 第一步, 先证明 $\boldsymbol{p}_k$ 为某个特定的最小二乘问题的极小残量.

**引理 10.2.2** *当 $k \geqslant 2$ 时, 由算法 (10.2.7) 产生的向量 $\boldsymbol{p}_k$ 满足*

$$\boldsymbol{p}_k = \boldsymbol{r}_{k-1} - \boldsymbol{A}\boldsymbol{P}_{k-1}\boldsymbol{z}_{k-1},$$

*其中 $\boldsymbol{P}_{k-1} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{k-1}]$, $\boldsymbol{z}_{k-1}$ 为*

$$\min_{\boldsymbol{z}\in\mathbb{R}^{k-1}} \| \boldsymbol{r}_{k-1} - \boldsymbol{A}\boldsymbol{P}_{k-1}\boldsymbol{z} \|_2$$

*的解.*

**证明** 设 $\boldsymbol{z}_{k-1}$ 为上面最小二乘问题的解, $\boldsymbol{p}$ 为相应的极小残量: $\boldsymbol{p} = \boldsymbol{r}_{k-1} - \boldsymbol{A}\boldsymbol{P}_{k-1}\boldsymbol{z}_{k-1}$. 由此可得 $\boldsymbol{p}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_{k-1} = 0$. 此外, $\boldsymbol{p} = [\boldsymbol{I} - (\boldsymbol{A}\boldsymbol{P}_{k-1})(\boldsymbol{A}\boldsymbol{P}_{k-1})^+]\boldsymbol{r}_{k-1}$ 是 $\boldsymbol{r}_{k-1}$ 到 $\mathrm{ran}(\boldsymbol{A}\boldsymbol{P}_{k-1})^{\perp}$ 上的正交投影, 故它是 $\mathrm{ran}(\boldsymbol{A}\boldsymbol{P}_{k-1})^{\perp}$ 中离 $\boldsymbol{r}_{k-1}$ 最近的向量. 因此 $p = p_k$. □

利用这一结果，我们可以得到残量 $\boldsymbol{r}_k$、搜索方向 $\boldsymbol{p}_k$ 和 Krylov 子空间

$$K(\boldsymbol{r}_0, \boldsymbol{A}, k) = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\}$$

之间的重要关系.

**定理 10.2.3** *在算法 (10.2.7) 中, 经过 $k$ 步迭代后有:*

$$\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k\boldsymbol{A}\boldsymbol{p}_k \tag{10.2.8}$$

$$\boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{r}_k = 0 \tag{10.2.9}$$

$$\mathrm{span}\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k\} = \mathrm{span}\{\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{k-1}\} = K(r_0, \boldsymbol{A}, k) \tag{10.2.10}$$

*且残量 $\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots, \boldsymbol{r}_k$ 互相正交.*

**证明** 在 $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k\boldsymbol{p}_k$ 两边左乘 $\boldsymbol{A}$ 并利用残量的定义可得方程 (10.2.8).

为证明 (10.2.9), 回想前面已得出 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{y}_k$, 其中 $\boldsymbol{y}_k$ 是

$$\phi(\boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{y}) = \phi(\boldsymbol{x}_0) + \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_k)\boldsymbol{y} - \boldsymbol{y}^{\mathrm{T}}\boldsymbol{P}_k(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$$

的极小点. 这意味着 $\boldsymbol{y}_k$ 为线性方程组 $(\boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_k)\boldsymbol{y} = \boldsymbol{P}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$ 的解, 因此

$$0 = \boldsymbol{P}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0) - \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{P}_k\boldsymbol{y}_k = \boldsymbol{P}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_0 + \boldsymbol{P}_k\boldsymbol{y}_k)) = \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{r}_k.$$

为证明 (10.2.10), 注意到从 (10.2.8) 有

$$\{\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}\} \subseteq \mathrm{span}\{\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{k-1}\},$$

而且从引理 10.2.2 有

$$\boldsymbol{p}_k = \boldsymbol{r}_{k-1} - [\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{A}\boldsymbol{p}_{k-1}]\boldsymbol{z}_{k-1} \in \mathrm{span}\{\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{k-1}\}.$$

从而

$$[\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k] = [\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{k-1}]\boldsymbol{T},$$

其中 $\boldsymbol{T}$ 为某个上三角形矩阵. 由于搜索方向是独立的, 故 $\boldsymbol{T}$ 非奇异. 这说明

$$\mathrm{span}\{\boldsymbol{p}_1,\cdots,\boldsymbol{p}_k\}=\mathrm{span}\{\boldsymbol{r}_0,\cdots,\boldsymbol{r}_{k-1}\}.$$

利用 (10.2.8) 可得

$$\boldsymbol{r}_k\in\mathrm{span}\{\boldsymbol{r}_{k-1},\boldsymbol{A}\boldsymbol{p}_k\}\subseteq\mathrm{span}\{\boldsymbol{r}_{k-1},\boldsymbol{A}\boldsymbol{r}_0,\cdots,\boldsymbol{A}\boldsymbol{r}_{k-1}\}.$$

由归纳法知 (10.2.10) 的后一等式成立.

最后, 证明这些残量是相互正交的. 从 (10.2.9) 可知 $\boldsymbol{r}_k$ 与 $\boldsymbol{p}_k$ 的象空间中的任一向量正交. 而由 (10.2.10) 可知此象空间包括 $\boldsymbol{r}_0,\boldsymbol{r}_1,\cdots,\boldsymbol{r}_{k-1}$. □

利用这些事实, 我们下一步证明 $\boldsymbol{p}_k$ 是前一个搜索方向 $\boldsymbol{p}_{k-1}$ 与当前残量 $\boldsymbol{r}_{k-1}$ 的线性组合.

**推论 10.2.4** (10.2.7) 中的残量和搜索方向有如下性质:

$$\boldsymbol{p}_k\in\mathrm{span}\{\boldsymbol{p}_{k-1},\boldsymbol{r}_{k-1}\},\quad k\geqslant 2$$

**证明** 当 $k=2$ 时, 由 (10.2.10) 可得 $\boldsymbol{p}_2\in\mathrm{span}\{\boldsymbol{r}_0,\boldsymbol{r}_1\}$. 但 $\boldsymbol{p}_1=\boldsymbol{r}_0$, 所以 $\boldsymbol{p}_2$ 是 $\boldsymbol{p}_1$ 与 $\boldsymbol{r}_1$ 的线性组合.

当 $k>2$ 时, 把引理 10.2.2 中的向量 $\boldsymbol{z}_{k-1}$ 作如下划分:

$$\boldsymbol{z}_{k-1}=\begin{bmatrix}\boldsymbol{w}\\ \mu\end{bmatrix}\begin{matrix}k-2\\ 1\end{matrix}$$

运用恒等式 $\boldsymbol{r}_{k-1}=\boldsymbol{r}_{k-2}-\alpha_{k-1}\boldsymbol{A}\boldsymbol{p}_{k-1}$, 有

$$\begin{aligned}\boldsymbol{p}_k&=\boldsymbol{r}_{k-1}-\boldsymbol{A}\boldsymbol{P}_{k-1}\boldsymbol{z}_{k-1}\\&=\boldsymbol{r}_{k-1}-\boldsymbol{A}\boldsymbol{P}_{k-2}\boldsymbol{w}-\mu\boldsymbol{A}\boldsymbol{p}_{k-1}\\&=\left(1+\frac{\mu}{\alpha_{k-1}}\right)\boldsymbol{r}_{k-1}+\boldsymbol{s}_{k-1},\end{aligned}$$

其中

$$\begin{aligned}\boldsymbol{s}_{k-1}&=-\frac{\mu}{\alpha_{k-1}}\boldsymbol{r}_{k-2}-\boldsymbol{A}\boldsymbol{P}_{k-2}\boldsymbol{w}\\&\in\mathrm{span}\{\boldsymbol{r}_{k-2},\boldsymbol{A}\boldsymbol{P}_{k-2}\boldsymbol{w}\}\\&\subseteq\mathrm{span}\{\boldsymbol{r}_{k-2},\boldsymbol{A}\boldsymbol{p}_1,\cdots,\boldsymbol{A}\boldsymbol{p}_{k-2}\}\\&\subseteq\mathrm{span}\{\boldsymbol{r}_1,\cdots,\boldsymbol{r}_{k-2}\}.\end{aligned}$$

因为 $\boldsymbol{r}_i$ 总是相互正交的, 故 $\boldsymbol{s}_{k-1}$ 和 $\boldsymbol{r}_{k-1}$ 正交. 因此, 引理 10.2.2 中的最小二乘问题等价于选择 $w$ 和 $\mu$, 使得

$$\|\boldsymbol{p}_k\|_2^2=\left(1+\frac{\mu}{\alpha_{k-1}}\right)^2\|\boldsymbol{r}_{k-1}\|_2^2+\|\boldsymbol{s}_{k-1}\|_2^2$$

达到极小. 既然 $\boldsymbol{r}_{k-2}-\boldsymbol{A}\boldsymbol{P}_{k-2}z$ 的 2 范数当 $z=z_{k-2}$ 时极小且残量为 $\boldsymbol{p}_{k-1}$, 所以 $\boldsymbol{s}_{k-1}$ 是 $\boldsymbol{p}_{k-1}$ 的倍数. 因此 $\boldsymbol{p}_k\in\mathrm{span}\{\boldsymbol{r}_{k-1},\boldsymbol{p}_{k-1}\}$ □

我们现在导出 $\boldsymbol{p}_k$ 的一个简单的表达式. 不失一般性, 根据推论 10.2.4 可设

$$\boldsymbol{p}_k=\boldsymbol{r}_{k-1}+\beta_k\boldsymbol{p}_{k-1}.$$

由 $\boldsymbol{p}_{k-1}^{\mathrm T}\boldsymbol{A}\boldsymbol{p}_k=0$, 可得

$$\beta_k=-\boldsymbol{p}_{k-1}^{\mathrm T}\boldsymbol{A}\boldsymbol{r}_{k-1}/\boldsymbol{p}_{k-1}^{\mathrm T}\boldsymbol{A}\boldsymbol{p}_{k-1}$$

这就得出共轭梯度法的“版本 1”:

$x_0 =$ 初值
$k = 0$
$r_0 = b - Ax_0$
**while** $r_k \neq 0$
  $k = k + 1$
  **if** $k = 1$
    $p_1 = r_0$
  **else**
    $\beta_k = -p_{k-1}^{\mathrm{T}} A r_{k-1} / p_{k-1}^{\mathrm{T}} A p_{k-1}$
    $p_k = r_{k-1} + \beta_k p_{k-1}$ (10.2.11)
  **end**
  $\alpha_k = p_k^{\mathrm{T}} r_{k-1} / p_k^{\mathrm{T}} A p_k$
  $x_k = x_{k-1} + \alpha_k p_k$
  $r_k = b - Ax_k$
**end**
$x = x_k$

在算法的运行当中，每步需要三次不同的矩阵与向量相乘. 然而，利用 $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{A}\boldsymbol{p}_k$ 递归地计算残量以及将

$$\boldsymbol{r}_{k-1}^{\mathrm{T}} \boldsymbol{r}_{k-1} = -\alpha_{k-1} \boldsymbol{r}_{k-1}^{\mathrm{T}} \boldsymbol{A}\boldsymbol{p}_{k-1}, \tag{10.2.12}$$

$$\boldsymbol{r}_{k-2}^{\mathrm{T}} \boldsymbol{r}_{k-2} = \alpha_{k-1} \boldsymbol{p}_{k-1}^{\mathrm{T}} \boldsymbol{A}\boldsymbol{p}_{k-1} \tag{10.2.13}$$

代入 $\beta_k$ 的表达式中，就得到下面更有效的形式.

**算法 10.2.1(共轭梯度法)** 如果 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 对称正定, $\boldsymbol{b} \in \mathbb{R}^n, \boldsymbol{x}_0 \in \mathbb{R}^n$ 为初始近似值 ($\boldsymbol{A}\boldsymbol{x}_0 \approx \boldsymbol{b}$), 则下列算法可计算出 $x \in \mathbb{R}^n$, 使得 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

$k = 0$
$r_0 = b - Ax_0$
**while** $r_k \neq 0$
  $k = k + 1$
  **if** $k = 1$
    $p_1 = r_0$
  **else**
    $\beta_k = r_{k-1}^{\mathrm{T}} r_{k-1} / r_{k-2}^{\mathrm{T}} r_{k-2}$
    $p_k = r_{k-1} + \beta_k p_{k-1}$
  **end**
  $\alpha_k = r_{k-1}^{\mathrm{T}} r_{k-1} / p_k^{\mathrm{T}} A p_k$
  $x_k = x_{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k A p_k$

**end**

$x = x_k$

这个算法本质上就是出现在 Hestenes and Stiefel(1952) 原文中的共轭梯度法的形式. 注意, 每步迭代只用到一次矩阵与向量相乘.

### 10.2.5 与 Lanczos 方法的联系

在 9.3.1 节中, 我们从 Lanczos 算法中导出了共轭梯度法. 现在, 我们通过从共轭梯度法"导出" Lanczos 方法来看两个算法之间的联系. 令残量矩阵为

$$\boldsymbol{R}_k = [r_0, r_1, \cdots, r_{k-1}] \in \mathbb{R}^{n\times k},$$

$\boldsymbol{B}_k$ 为如下的上双对角矩阵

$$\boldsymbol{B}_k = \begin{bmatrix} 1 & -\beta_2 & 0 & \cdots & 0 \\ 0 & 1 & -\beta_3 & & \vdots \\ & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & -\beta_k \\ 0 & \cdots & & 0 & 1 \end{bmatrix} \in \mathbb{R}^{k\times k}.$$

从方程 $\boldsymbol{p}_i = \boldsymbol{r}_{i-1} + \beta_i \boldsymbol{p}_{i-1}, i = 2:k, \boldsymbol{p}_1 = \boldsymbol{r}_0$ 可知 $\boldsymbol{R}_k = \boldsymbol{P}_k\boldsymbol{B}_k$. 既然矩阵 $\boldsymbol{P}_k = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k]$ 的列是 A 共轭的, 我们可以看出 $\boldsymbol{R}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{R}_k = \boldsymbol{B}_k^{\mathrm{T}}\mathrm{diag}(\boldsymbol{p}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_1, \cdots, \boldsymbol{p}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_k)\boldsymbol{B}_k$ 为三对角矩阵. 从 (10.2.10) 可知, 如果

$$\boldsymbol{\Delta} = \mathrm{diag}(\rho_0, \cdots, \rho_{k-1}), \quad \rho_i = \| \boldsymbol{r}_i \|_2,$$

则 $\boldsymbol{R}_k\boldsymbol{\Delta}^{-1}$ 的列构成子空间 $\mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots \boldsymbol{A}^{k-1}\boldsymbol{r}_0\}$. 的一组标准正交基. 因此, 这个矩阵的列本质上是算法 9.3.1 中的 Lanczos 向量, 即

$$\boldsymbol{q}_i = \pm \boldsymbol{r}_{i-1}/\rho_{i-1}, \quad i = 1:k.$$

此外, 相应于这些 Lanczos 向量的三对角矩阵为

$$\boldsymbol{T}_k = \boldsymbol{\Delta}^{-1}\boldsymbol{B}_k^{\mathrm{T}}\mathrm{diag}(\boldsymbol{p}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{p}_i)\boldsymbol{B}_k\boldsymbol{\Delta}^{-1}. \tag{10.2.14}$$

这个矩阵的主对角元和次对角元在共轭梯度迭代中是可以得到的. 因此, 在算法 10.2.1 中, 产生 $\boldsymbol{x}_k$ 的同时, 也得到了 $\boldsymbol{A}$ 的两端的特征值 (条件数) 很好的估计.

### 10.2.6 一些实用的细节

算法 10.2.1 的终止准则是不现实的. 由于舍入误差的影响, 残量间的正交性有所损失, 从而在数学上不能保证有有限步终止. 此外, 当应用共轭梯度法时, $n$ 通常非常大, 使得 $O(n)$ 步迭代的工作量也是不可接受的. 因此, 通常根据极大迭代次数 $k_{\max}$ 以及残量的范数来给出终止准则. 这就是导出算法 10.2.1 的实用形式:

$x$=初值

$k = 0$

$r = b - Ax_0$

$$
\begin{aligned}
&\rho_0 = \| r \|_2^2 \\
&\textbf{while } (\sqrt{\rho_k} > \varepsilon \| b \|_2) \wedge (k < k_{\max}) \\
&\qquad k = k+1 \\
&\qquad \textbf{if } k = 1 \\
&\qquad\qquad p = r \\
&\qquad \textbf{else} \\
&\qquad\qquad \beta_k = \rho_{k-1}/\rho_{k-2} \\
&\qquad\qquad p = r + \beta_k p \\
&\qquad \textbf{end} \\
&\qquad w = Ap \\
&\qquad \alpha_k = \rho_{k-1}/p^{\mathrm{T}} w \\
&\qquad x = x + \alpha_k p \\
&\qquad r = r - \alpha_k w \\
&\qquad \rho_k = \| r \|_2^2 \\
&\textbf{end}
\end{aligned}
\tag{10.2.15}
$$

在每步迭代中, 这一算法需要一次矩阵与向量相乘和 $10n$ 次浮点运算. 注意, 实际上只要存储四个向量: $\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{p}$ 和 $\boldsymbol{w}$. 标量的下标是不必要的, 这里不过是用来与算法 10.2.1 比较而已.

用 (10.2.14) 给出的三对角矩阵 $\boldsymbol{T}_k$ 的最小特征值的倒数来近似 $\| \boldsymbol{A}^{-1} \|_2$, 可得到误差 $\boldsymbol{A}^{-1}\boldsymbol{r}_k$ 的启发式估计, 可基于此给出终止准则.

把共轭梯度法看成一种迭代法的思想是 Reid(1971) 最先提出的. 迭代的观点是很有意义的, 但是收敛的速度则是此方法成功的关键.

### 10.2.7 收敛的性质

作为本节的结尾, 我们考察一下共轭梯度法产生的迭代序列 $\{x_k\}$ 的收敛性. 我们给出两个结果, 它们都说明了不管是在低秩扰动的意义下还是在范数的意义下, 当 $\boldsymbol{A}$ 接近于单位矩阵时, 共轭梯法非常有效.

**定理 10.2.5** *如果 $\boldsymbol{A} = \boldsymbol{I} + \boldsymbol{B}$ 为 $n \times n$ 对称正定矩阵, 且 $\mathrm{rank}(\boldsymbol{B}) = r$, 则算法 10.2.1 至多 $r+1$ 步收敛.*

**证明** 子空间 $\mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, \cdots, \boldsymbol{A}^{k-1}\boldsymbol{r}_0\} = \mathrm{span}\{\boldsymbol{r}_0, \boldsymbol{B}\boldsymbol{r}_0, \cdots, \boldsymbol{B}^{k-1}\boldsymbol{r}_0\}$ 的维数不超过 $r+1$. 既然 $p_1, \cdots, p_k$ 可生成这一子空间, 且线性独立, 则迭代次数不能大于 $r+1$. □

从这一结果可得如下重要的结论:

- *如果 $\boldsymbol{A}$ 接近于单位矩阵的秩 $r$ 校正, 则算法 10.2.1 经过 $k+1$ 步后差不多就可收敛.*

在下一节中, 我们将说明如何利用此直观结果.

另一种形式的误差界可用 $A$ 范数来得到, 定义 $A$ 范数:

$$\| w \|_A = \sqrt{w^T A w}.$$

**定理 10.2.6** 假设 $A \in \mathbb{R}^{n \times n}$ 对称正定, $b \in \mathbb{R}^n$. 若 $\{x_k\}$ 为算法 10.2.1 产生的迭代序列, $\kappa = \kappa_2 A$, 则

$$\| x - x_k \|_A \leqslant 2 \| x - x_0 \|_A \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k.$$

**证明** 见 Luenberger(1973, 187 页). □

$\{x_k\}$ 的精度经常比定理估计的要好得多. 然而, 定理 10.2.6 的一个探试形式倒很有用:

- 若 $\kappa_2(A) \approx 1$, 则在 $A$ 范数意义下, 共轭梯度法收敛很快.

在下一节中, 我们论述如何把一个给定的问题 $Ax = b$ 转换成一相关的问题 $\tilde{A} \tilde{x} = \tilde{b}$, 其中 $\tilde{A}$ 接近于单位矩阵.

## 习 题

**10.2.1** 证明 (10.2.1) 中的残量满足: 当 $j = i + 1$ 时 $r_i^T r_j = 0$.

**10.2.2** 证明 (10.2.2).

**10.2.3** 证明 (10.2.3).

**10.2.4** 证明 (10.2.12) 和 (10.2.13).

**10.2.5** 给出 (10.2.14) 中三对角矩阵 $T_k$ 的元素的计算公式.

**10.2.6** 在算法 9.3.1 和算法 10.2.1 的实际实现中, 比较它们的计算量和存储量.

**10.2.7** 证明: 若 $A \in \mathbb{R}^{n \times n}$ 是对称正定的, 且有 $k$ 个相异的特征值, 则共轭梯度法最多运行 $k + 1$ 步就结束.

**10.2.8** 利用定理 10.2.6 证明:

$$\| x_k - A^{-1} b \|_2 \leqslant 2\sqrt{\kappa} \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \| x_0 - A^{-1} b \|_2 .$$

## 本节注释与参考文献

共轭梯度法是一大类方法 ——*共轭方向*算法中的一种, 在共轭方向算法中, 所有的搜索方向是 B 共轭的, 对某一合适的矩阵 $B$. 这些方法的讨论可见:

J. E. Dennis Jr. and K. Turner (1987). "Generalized Conjugate Directions," *Lin. Alg. and Its Applic.* 88/89, 187–209.

G. W. Stewart (1973). "Conjugate Direction Methods for Solving Systems of Linear Equations," *Numer. Math.* 21, 284–297.

一些经典的和综合性的讨论可见:

G. Golub and D. O'Leary (1989). "Some History of the Conjugate Gradient and Lanczos Methods," *SIAM Review* 31, 50–102.

M. R. Hestenes (1990). "Conjugacy and Gradients," in *A History of Scientific Computing,* Addison-Wesley, Reading, MA.

S. Ashby, T.A. Manteuffel. and P. E. Saylor (1992). "A Taxonomy for Conjugate Gradient Methods," *SIAM J. Numer. Anal.* 27, 1542–1568.

共轭梯度法的经典文献可见:

M. R. Hestenes and E. Stiefel (1952). "Methods of Conjugate Gradients for Solving Linear Systems," *J. Res. Nat. Bur. Stand.* 49, 409–436.

这个方法的精确算术分析见下文的第 2 章:

M. R. Hestenes (1980). *Conjugate Direction Methods in Optimization*, Springer-Verlag, Berlin.

也见:

O. Axelsson (1977). "Solution of Linear Systems of Equations: Iterative Methods," in *Sparse Matrix Techniques: Copenhagen*, 1976, ed. V. A. Barker, Springer-Verlag, Berlin.

关于共轭梯度法收敛表现的讨论可见:

D. G. Luenberger (1973). *Introduction to Linear and Nonlinear programming*, Addison-Wesley, New York.

A. van der Sluis and H. A. Van Der Vorst (1986). "The Rate of Convergence of Conjugate Gradients," *Numer. Math.* 48, 543–560.

把共轭梯度法作为迭代法的思想首先出现在:

J. K. Reid (1971). "On the Method of Conjugate Gradients for the Solution of Large Sparse Systems of Linear Equations," *in Large Sparse Sets of Linear Equations*, ed. J. K. Reid, Academic Press, New York, pp. 231–254.

有一些作者试图用有限精度的算述运算来解释算法的表现, 见:

H. Wozniakowski (1980). "Roundoff Error Analysis of a New Class of Conjugate Gradient Algorithms," *Lin. Alg. and Its Applic.* 29.

A. Greenbaum and Z. Strakos (1992). "Predicting the Behavior of Finite Precision lanczos and Conjugate Gradient Computations," *SIAM J. Matrix Ana. Applic.* 13, 121–137.

也见下文的分析:

G. W. Stewart (1975). "The Convergence of the Method of Conjugate Gradients at Isolated Extreme Points in the Spectrum," *Numer. Math.* 24, 85–93.

A. Jennings (1977). "Influence of the Eigenvalue Spectrum on the Convergence Rate of the Conjugate Gradient Method," *J. Inst. Math. Applic.* 20, 61–72.

J. Cullum and R. Willoughby (1980). "The Lanczos Phenomena: An Interpretation Based on Conjugate Gradient Optimization," *Lin. Alg. and Its Applic.* 29, 63–90.

最后, 我们提一下, 这方法可用来计算大型稀疏对称矩阵的特征向量.

A. Ruhe and T. Wiberg (1972). "The Method of Conjugate Gradients Used in Inverse Iteration," *BIT* 12, 543–554.

## 10.3 预处理共轭梯度

我们在上一节的结尾提到, 当矩阵为良态或有少数几个不同的特征值时, 共轭梯度法用起来很有效. (后一种情况即 $\boldsymbol{A}$ 是单位矩阵的低秩扰动.) 本节中, 我们将介绍如何预处理线性方程组, 使其系数矩阵为上述的两种情况之一. 我们的论述是

简洁、非正式的. Golub and Meurant (1983) 和 Axelsson (1985) 有更全面的阐述.

### 10.3.1 推导

考虑到 $n \times n$ 对称正定线性方程组 $\boldsymbol{Ax} = \boldsymbol{b}$. 预处理共轭梯度法的思想是把"常规的"共轭梯度法运用到变换之后的方程组:

$$\tilde{\boldsymbol{A}}\,\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}, \tag{10.3.1}$$

其中 $\tilde{\boldsymbol{A}} = \boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}^{-1}, \tilde{\boldsymbol{x}} = \boldsymbol{C}\boldsymbol{x}, \tilde{\boldsymbol{b}} = \boldsymbol{C}^{-1}\boldsymbol{b}$, $\boldsymbol{C}$ 为对称正定矩阵. 依照 10.2.8 节的结论, 我们试图选择矩阵 $\boldsymbol{C}$, 使得 $\tilde{\boldsymbol{A}}$ 是良态的或具有多重特征值. 不久将会看出, 矩阵 $\boldsymbol{C}^2$ 也必须简单一些.

如果把算法 10.2.1 用到 (10.3.1), 则得到:

$$
\begin{aligned}
&k = 0\\
&\tilde{x}_0 = \text{初值}(\tilde{A}\tilde{x}_0 \approx \tilde{b})\\
&\tilde{r}_0 = \tilde{b} - \tilde{A}\tilde{x}_0\\
&\textbf{while } \tilde{r}_k \neq 0\\
&\qquad k = k+1\\
&\qquad \textbf{if } k = 1\\
&\qquad\qquad \tilde{p}_1 = \tilde{r}_0\\
&\qquad \textbf{else}\\
&\qquad\qquad \beta_k = \tilde{r}_{k-1}^{\mathrm{T}}\tilde{r}_{k-1}/\tilde{r}_{k-2}^{\mathrm{T}}\tilde{r}_{k-2}\\
&\qquad\qquad \tilde{p}_k = \tilde{r}_{k-1} + \beta_k\tilde{p}_{k-1}\\
&\qquad \textbf{end}\\
&\qquad \alpha_k = \tilde{r}_{k-1}^{\mathrm{T}}\tilde{r}_{k-1}/\tilde{p}_k^{\mathrm{T}}C^{-1}AC^{-1}\tilde{p}_k\\
&\qquad \tilde{x}_k = \tilde{x}_{k-1} + \alpha_k\tilde{p}_k\\
&\qquad \tilde{r}_k = \tilde{r}_{k-1} - \alpha_k C^{-1}AC^{-1}\tilde{p}_k\\
&\textbf{end}\\
&\tilde{x} = \tilde{x}_k
\end{aligned}
\tag{10.3.2}
$$

这里, $\tilde{\boldsymbol{x}}_k$ 为 $\tilde{\boldsymbol{x}}$ 的近似, $\tilde{\boldsymbol{r}}_k$ 是在变换后的坐标系中的残量, 即 $\tilde{\boldsymbol{r}}_k = \tilde{\boldsymbol{b}} - \tilde{\boldsymbol{A}}\,\tilde{\boldsymbol{x}}_k$. 当然, 一旦得到 $\tilde{\boldsymbol{x}}$ 的值, 则通过方程 $\boldsymbol{x} = \boldsymbol{C}^{-1}\tilde{\boldsymbol{x}}$ 可得到 $\boldsymbol{x}$ 的值. 然而利用定义 $\tilde{p}_k = \boldsymbol{C}p_k, \tilde{\boldsymbol{x}}_k = \boldsymbol{C}\boldsymbol{x}_k$ 及 $\tilde{\boldsymbol{r}}_k = \boldsymbol{C}^{-1}\boldsymbol{r}_k$, 就可避免直接利用矩阵 $\boldsymbol{C}^{-1}$. 实际上, 如果我们把上述定义用到算法 (10.3.2) 中, 并注意到关系式 $\tilde{\boldsymbol{b}} = \boldsymbol{C}^{-1}\boldsymbol{b}$ 和 $\tilde{\boldsymbol{x}} = \boldsymbol{C}\boldsymbol{x}$, 则得到下面的算法:

$$
\begin{aligned}
&k = 0\\
&x_0 = \text{初值}(Ax_0 \approx b)\\
&r_0 = b - Ax_0\\
&\textbf{while } C^{-1}r_k \neq 0\\
&\qquad k = k+1
\end{aligned}
$$

$$
\begin{aligned}
&\quad \textbf{if } k=1 \\
&\qquad Cp_1 = C^{-1}r_0 \\
&\quad \textbf{else} \\
&\qquad \beta_k = (C^{-1}r_{k-1})^{\mathrm{T}}(C^{-1}r_{k-1})/(C^{-1}r_{k-2})^{\mathrm{T}}(C^{-1}r_{k-2}) \\
&\qquad Cp_k = C^{-1}r_{k-1} + \beta_k Cp_{k-1} \\
&\quad \textbf{end} \\
&\quad \alpha_k = (C^{-1}r_{k-1})^{\mathrm{T}}(C^{-1}r_{k-1})/(Cp_k)^{\mathrm{T}}(C^{-1}AC^{-1})(Cp_k) \\
&\quad Cx_k = Cx_{k-1} + \alpha_k Cp_k \\
&\quad C^{-1}r_k = C^{-1}r_{k-1} - \alpha_k(C^{-1}AC^{-1})Cp_k \\
&\textbf{end} \\
&Cx = Cx_k
\end{aligned}
\tag{10.3.3}
$$

如果我们定义预处理矩阵 $\boldsymbol{M} = \boldsymbol{C}^2$(也是正定的), 且令 $\boldsymbol{z}_k$ 为方程组 $\boldsymbol{M}\boldsymbol{z}_k = \boldsymbol{r}_k$ 的解, 则算法 (10.3.3) 可简写如下.

**算法 10.3.1(预处理共轭梯度法)** 给定对称正定矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}, \boldsymbol{b}\in\mathbb{R}^n$, 对称正定的预处理矩阵 $\boldsymbol{M}$ 和初值 $\boldsymbol{x}_0(\boldsymbol{A}\boldsymbol{x}_0\approx\boldsymbol{b})$, 则本算法可求出方程组 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 的解.

$$
\begin{aligned}
&k = 0 \\
&r_0 = b - Ax_0 \\
&\textbf{while } (r_k \neq 0) \\
&\quad \text{解方程 } Mz_k = r_k \\
&\quad k = k+1 \\
&\quad \textbf{if } k=1 \\
&\qquad p_1 = z_0 \\
&\quad \textbf{else} \\
&\qquad \beta_k = r_{k-1}^{\mathrm{T}}z_{k-1}/r_{k-2}^{\mathrm{T}}z_{k-2} \\
&\qquad p_k = z_{k-1} + \beta_k p_{k-1} \\
&\quad \textbf{end} \\
&\quad \alpha_k = r_{k-1}^{\mathrm{T}}z_{k-1}/p_k^{\mathrm{T}}Ap_k \\
&\quad x_k = x_{k-1} + \alpha_k p_k \\
&\quad r_k = r_{k-1} - \alpha_k Ap_k \\
&\textbf{end} \\
&x = x_k
\end{aligned}
$$

关于这一算法, 应注意到以下重要的几点.

- 可以证明, 残量和搜索方向满足:

$$
\boldsymbol{r}_j^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{r}_i = 0, \quad i \neq j, \tag{10.3.4}
$$

$$
\boldsymbol{p}_j^{\mathrm{T}}(\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}^{-1})\boldsymbol{p}_i = 0, \quad i \neq j, \tag{10.3.5}
$$

- 分母 $r_{k-2}^{\mathrm{T}}z_{k-2}=z_{k-2}^{\mathrm{T}}Mz_{k-2}$ 不可能为零, 因为 $M$ 是正定的.
- 虽然变换矩阵 $C$ 在算法的导出中起了很重要的作用, 但它只有通过预处理矩阵 $M=C^2$ 起作用.
- 为了使算法 10.3.1 在处理稀疏矩阵时有效, 线性方程组 $Mz=r$ 必须容易求解, 且收敛速度还要快.

一个好的预处理矩阵对算法的收敛速度产生很大的影响. 现讨论几种可能的预处理矩阵.

### 10.3.2 不完全 Cholesky 预处理矩阵

最重要的预处理方式之一涉及计算 $A$ 的不完全 Cholesky 分解. 其思想是计算一下三角形矩阵 $H$, 它具有可利用的稀疏结构, 且"接近于" $A$ 的完全 Cholesky 因子 $G$. 预处理矩阵则变为 $M=HH^{\mathrm{T}}$. 为说明这一选择的作用, 先注意下列事实.

- 存在唯一的对称正定矩阵 $C$, 使得 $M=C^2$
- 存在正定矩阵 $Q$, 使得 $C=QH^{\mathrm{T}}$, 即 $H^{\mathrm{T}}$ 是 $C$ 的 QR 分解的上三角形因子.

由此可知:

$$\begin{aligned}\tilde{A}&=C^{-1}AC^{-1}=C^{-\mathrm{T}}AC^{-1}=(HQ^{\mathrm{T}})^{-1}A(QH^{\mathrm{T}})^{-1}\\&=Q(H^{-1}GG^{\mathrm{T}}H^{-\mathrm{T}})Q^{\mathrm{T}}\approx I.\end{aligned}\tag{10.3.6}$$

因此, $H$ 逼近 $G$ 越好, $\tilde{A}$ 的条件数越小, 算法 10.3.1 运行效果就越好.

如何找到这样的 $H$, 一种简单而又有效的方法是在 Cholesky 分解中, 若 $a_{ij}=0$, 则令 $h_{ij}=0$. 若 Cholesky 分解采用外积形式, 上述方法可如下实现:

```
for k = 1 : n
    A(k,k) = sqrt(A(k,k))
    for i = k+1 : n
        if A(i,k) ≠ 0
            A(i,k) = A(i,k)/A(k,k)                         (10.3.7)
        end
    end
    for j = k+1 : n
        for i = j : n
            if A(i,j) ≠ 0
                A(i,j) = A(i,j) - A(i,k)A(j,k)
            end
        end
    end
end
```

在实际计算中, 矩阵 $\boldsymbol{A}$ 和它的不完全 Cholesky 因子 $\boldsymbol{H}$ 可以存储在一个合适的数据结构中, 上述算法中的循环将呈现特殊的形式.

不幸的是, (10.3.7) 并不总是稳定的. Manteuffel (1979) 指出了不完全 Cholesky 分解稳定的正定矩阵类. 也可参见 Elman(1986).

### 10.3.3 不完全分块预处理矩阵

和本书中所有其他方法一样, 上一小节中不完全分解的思想也有块的类似形式. 我们以对称正定的分块三对角矩阵

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{E}_1^{\mathrm{T}} & \boldsymbol{0}\\ \boldsymbol{E}_1 & \boldsymbol{A}_2 & \boldsymbol{E}_2^{\mathrm{T}}\\ \boldsymbol{0} & \boldsymbol{E}_2 & \boldsymbol{A}_3\end{bmatrix}$$

为例来说明不完全分块 Cholesky 分解. 为了说明问题, 我们假设 $\boldsymbol{A}_i$ 是三对角的, $\boldsymbol{E}_i$ 为对角的. 二维区域上的自伴椭圆型偏微分方程用标准的五点格式离散化以后得到的就是这种结构的矩阵.

$3\times 3$ 的块是很普遍的. 我们的讨论基于 Concus , Golub, and Meurant(1985). 令

$$\boldsymbol{G}=\begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{F}_1 & \boldsymbol{G}_2 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{F}_2 & \boldsymbol{G}_3\end{bmatrix}$$

为 $\boldsymbol{A}$ 的完全 Cholesky 因子的分块形式. 虽然 $\boldsymbol{G}$ 作为分块矩阵在整体上是稀疏的, 但除 $\boldsymbol{G}_1$ 外的每块都是稠密的. 这可以从相应的计算中看出:

$$\begin{aligned}\boldsymbol{G}_1\boldsymbol{G}_1^{\mathrm{T}}&=\boldsymbol{B}_1\equiv\boldsymbol{A}_1,\\ \boldsymbol{F}_1&=\boldsymbol{E}_1\boldsymbol{G}_1^{-1},\\ \boldsymbol{G}_2\boldsymbol{G}_2^{\mathrm{T}}&=\boldsymbol{B}_2\equiv\boldsymbol{A}_2-\boldsymbol{F}_1\boldsymbol{F}_1^{\mathrm{T}}=\boldsymbol{A}_2-\boldsymbol{E}_1\boldsymbol{B}_1^{-1}\boldsymbol{E}_1^{\mathrm{T}},\\ \boldsymbol{F}_2&=\boldsymbol{E}_2\boldsymbol{G}_2^{-1},\\ \boldsymbol{G}_3\boldsymbol{G}_3^{\mathrm{T}}&=\boldsymbol{B}_3\equiv\boldsymbol{A}_3-\boldsymbol{F}_2\boldsymbol{F}_2^{\mathrm{T}}=\boldsymbol{A}_3-\boldsymbol{E}_2\boldsymbol{B}_2^{-1}\boldsymbol{E}_2^{\mathrm{T}}.\end{aligned}$$

因此, 我们要寻找近似的分块 Cholesky 因子, 其形式为

$$\widetilde{\boldsymbol{G}}=\begin{bmatrix}\widetilde{\boldsymbol{G}}_1 & \boldsymbol{0} & \boldsymbol{0}\\ \widetilde{\boldsymbol{F}}_1 & \widetilde{\boldsymbol{G}}_2 & \boldsymbol{0}\\ \boldsymbol{0} & \widetilde{\boldsymbol{F}}_2 & \widetilde{\boldsymbol{G}}_3\end{bmatrix},$$

从而, 我们能够容易地求解预处理矩阵为 $\boldsymbol{M}=\widetilde{\boldsymbol{G}}\widetilde{\boldsymbol{G}}^{\mathrm{T}}$ 的方程组. 这用到了 $\widetilde{\boldsymbol{G}}$ 中块的稀疏性. 当 $\boldsymbol{A}_i$ 是三对角矩阵, $\boldsymbol{E}_i$ 为对角矩阵时, 有下面一条合理的途径:

$$\begin{aligned}\widetilde{\boldsymbol{G}}_1\widetilde{\boldsymbol{G}}_1^{\mathrm{T}}&=\widetilde{\boldsymbol{B}}_1\equiv\boldsymbol{A}_1,\\ \widetilde{\boldsymbol{F}}_1&=\boldsymbol{E}_1\widetilde{\boldsymbol{G}}_1^{-1},\end{aligned}$$

$$\begin{aligned}\widetilde{G}_2\widetilde{G}_2^{\mathrm{T}} &= \widetilde{B}_2 \equiv A_2 - E_1\Lambda_1E_1^{\mathrm{T}}, \Lambda_1(\text{三对角}) \approx \widetilde{B}_1^{-1}\\ \widetilde{F}_2 &= E_2\widetilde{G}_2^{-1}.\\ \widetilde{G}_3\widetilde{G}_3^{\mathrm{T}} &= \widetilde{B}_3 \equiv A_3 - E_2\Lambda_2E_2^{\mathrm{T}}, \Lambda_2(\text{三对角}) \approx \widetilde{B}_2^{-1}.\end{aligned}$$

注意, 所有的 $\widetilde{B}_i$ 都是三对角的. 显然, 必须小心地选择 $\Lambda_i$, 使得 $\widetilde{B}_i$ 也是对称正定的. 由此可得 $\widetilde{G}_i$ 为下双对角矩阵. $\widetilde{F}_i$ 是满的, 但不需要显式地计算出它的元素. 比如, 在解方程组 $Mz = r$ 的过程中, 我们必须解如下方程:

$$\begin{bmatrix} \widetilde{G}_1 & \mathbf{0} & \mathbf{0} \\ \widetilde{F}_1 & \widetilde{G}_2 & \mathbf{0} \\ \mathbf{0} & \widetilde{F}_2 & \widetilde{G}_3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}.$$

在计算涉及关于 $\widetilde{F}_i = E_i\widetilde{G}_i^{-1}$ 的矩阵与向量相乘时, 可用向前消去法:

$$\begin{aligned}\widetilde{G}_1w_1 &= r_1,\\ \widetilde{G}_2w_2 &= r_2 - \widetilde{F}_1w_1 = r_2 - E_1\widetilde{G}_1^{-1}w_1,\\ \widetilde{G}_3w_3 &= r_3 - \widetilde{F}_2w_2 = r_3 - E_2\widetilde{G}_2^{-1}w_2.\end{aligned}$$

为了保证 $\widetilde{B}_i$ 的正定性, $\Lambda_i$ 的选取是比较精细的. 和我们所组织的计算一样, 核心问题是如何用一个对称的三对角矩阵 $\Lambda$ 来近似 $m \times m$ 对称正定的三对角矩阵 $T = (t_{ij})$ 的逆. 这里有几条合理的途径:

- 令 $\Lambda = \mathrm{diag}(1/t_{11}, 1/t_{22}, \cdots, 1/t_{nn})$.
- 令 $\Lambda$ 为 $T^{-1}$ 的三对角部分. 这可以很有效地计算出来, 因为存在 $u, v \in \mathbb{R}^m$ 使得 $T^{-1}$ 的下三角部分正好是 $uv^{\mathrm{T}}$ 的下三角部分. 见 Asplund(1959).
- 令 $\Lambda = U^{\mathrm{T}}U$, $U$ 为 $G^{-1}$ 的下双对角部分, 其中 $G$ 为 $T$ 的 Cholesky 分解因子, 即 $T = GG^{\mathrm{T}}$. 这可用 $O(m)$ 个 flop 来完成.

要讨论这些近似矩阵及相应的预处理矩阵之性质, 可参见 Concus, Golub, and Meurant(1985).

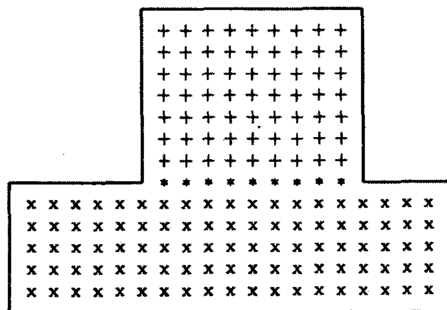
### 10.3.4 区域分解思想

椭圆型偏微分方程的数值求解在未知数适当排序时往往导致如下方程组:

$$\begin{bmatrix} A_1 & \cdots & & \cdots & B_1 \\ \vdots & A_2 & & & B_2 \\ & & \ddots & & \vdots \\ \vdots & & & A_p & B_p \\ B_1^{\mathrm{T}} & B_2^{\mathrm{T}} & \cdots & B_p^{\mathrm{T}} & Q \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \\ z \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_p \\ f \end{bmatrix}. \tag{10.3.8}$$

见 Meurant (1984). 这里 $A_i$ 是对称正定的, $B_i$ 是稀疏的, 最后一块列的列数比其他的要少得多.

$p=2$ 的例子可说明 (10.3.8) 及其分块结构与原问题几何区域及其分解之间的联系. 假设在下面的区域上解 Poisson 方程:



用通常的方法离散, 一个网格点上的未知量只与"东南西北"四点相关. 这里有三种类型的变量: 一类是内部在上面的子区域 (聚集了子向量 $x_1$, 相应的网格点为"+"), 另一类是内部在下面的子区域 (聚集了子向量 $\boldsymbol{x}_2$, 相应的网格点为"x"), 还有一类是在这两类子区域的交界处 (聚集了子向量 $\boldsymbol{z}$, 相应的网格点为"*"). 注意, 一个子区域的内部未知量与另一子区域的内部未知量无关, 这说明了 (10.3.8) 中为什么会出现零块. 同样可以观察到, 交界处的未知量的个数比整个未知量的个数要少得多.

现在, 我们来探索一下 (10.3.8) 的预处理情况. 为简单起见, 就 $p=2$ 的情形讨论. 令

$$\boldsymbol{M}=\boldsymbol{L}\begin{bmatrix}\boldsymbol{M}_1^{-1} & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{M}_2^{-1} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{S}^{-1}\end{bmatrix}\boldsymbol{L}^{\mathrm{T}},\quad \text{其中}\boldsymbol{L}=\begin{bmatrix}\boldsymbol{M}_1 & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{M}_2 & \boldsymbol{0}\\ \boldsymbol{B}_1^{\mathrm{T}} & \boldsymbol{B}_2^{\mathrm{T}} & \boldsymbol{S}\end{bmatrix},$$

则

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{M}_1 & \boldsymbol{0} & \boldsymbol{B}_1\\ \boldsymbol{0} & \boldsymbol{M}_2 & \boldsymbol{B}_2\\ \boldsymbol{B}_1^{\mathrm{T}} & \boldsymbol{B}_2^{\mathrm{T}} & \boldsymbol{S}_*\end{bmatrix}, \tag{10.3.9}$$

这里 $\boldsymbol{S}_*=\boldsymbol{S}+\boldsymbol{B}_1^{\mathrm{T}}\boldsymbol{M}_1^{-1}\boldsymbol{B}_1+\boldsymbol{B}_2^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{B}_2$. 现考虑如何选择块参数 $\boldsymbol{M}_1$, $\boldsymbol{M}_2$ 和 $\boldsymbol{S}$, 使得产生一个有效的预处理矩阵.

如果把 (10.3.9) 和 (10.3.8) 在 $p=2$ 时的情形相比, 可以看出, 应该用 $\boldsymbol{M}_i$ 逼近 $\boldsymbol{A}_i$, 用 $\boldsymbol{S}_*$ 逼近 $\boldsymbol{Q}$. 若令 $\boldsymbol{S}\approx\boldsymbol{Q}-\boldsymbol{B}_1^{\mathrm{T}}\boldsymbol{M}_1^{-1}\boldsymbol{B}_1-\boldsymbol{B}_2^{\mathrm{T}}\boldsymbol{M}_2^{-1}\boldsymbol{B}_2$, 则后者可以办到. 选取 $\boldsymbol{S}$ 有好几种途径, 但都要注意这样一个事实: 不能有稠密的矩阵 $\boldsymbol{B}_i\boldsymbol{M}_i^{-1}\boldsymbol{B}_i^{\mathrm{T}}$ 产生. 例如, 在上一节讨论过, 可用 $\boldsymbol{M}_i^{-1}$ 的三对角近似, 见 Meurant(1989).

如果子区域足够规则且解只与 $\boldsymbol{A}_i$ 有关的方程是可行的话 (如用快速 Poisson 解法), 则可令 $\boldsymbol{M}_i=\boldsymbol{A}_i$. 从而 $\boldsymbol{M}=\boldsymbol{A}+\boldsymbol{E}$, 其中 $\mathrm{rank}(\boldsymbol{E})=m$, $m$ 为交界处未知量的个数. 因此从理论上讲, 经过 $m+1$ 步以后, 预处理共轭梯度法将收敛.

不考虑算法过程中必须进行的逼近, 我们有很多并行计算的重大机遇, 因为子区域的问题是可分离的. 实际上, 子区域的个数 $p$ 通常与问题的几何特性以及可用来计算的处理器个数有关.

### 10.3.5 多项式预处理矩阵

由预处理矩阵 $\boldsymbol{M}$ 组成的方程组 $\boldsymbol{M}\boldsymbol{z}=\boldsymbol{r}$ 的解所定义的向量 $\boldsymbol{z}$ 应当看作是 $\boldsymbol{A}\boldsymbol{z}=\boldsymbol{r}$ 的一个近似解, 因为 $\boldsymbol{M}$ 是 $\boldsymbol{A}$ 的一个近似. 获得这个近似解的一种方法是用平稳方法 $\boldsymbol{M}_1\boldsymbol{z}^{(k+1)}=\boldsymbol{N}_1\boldsymbol{z}^{(k)}+\boldsymbol{r}$ 和 $\boldsymbol{z}^{(0)}=\mathbf{0}$ 的 $p$ 步.

从 $\boldsymbol{G}=\boldsymbol{M}_1^{-1}\boldsymbol{N}_1$ 可知

$$z\equiv z^{(p)}=(\boldsymbol{I}+\boldsymbol{G}+\cdots+\boldsymbol{G}^{p-1})\boldsymbol{M}_1^{-1}\boldsymbol{r}.$$

于是, 如果 $\boldsymbol{M}^{-1}=(\boldsymbol{I}+\boldsymbol{G}+\cdots+\boldsymbol{G}^{p-1})\boldsymbol{M}_1^{-1}$, 则 $\boldsymbol{M}z=r$, 我们可认为 $\boldsymbol{M}$ 是一个预处理矩阵. 当然, 重要的是 $\boldsymbol{M}$ 应当是对称正定的, 这限制了 $\boldsymbol{M}_1$, $\boldsymbol{N}_1$ 和 $p$ 的选取. 因为 $\boldsymbol{M}$ 是 $\boldsymbol{G}$ 的多项式, 故称它为*多项式预处理矩阵*. 这类*预处理矩阵*从向量或并行的观点看很有吸引力, 因而引起了广泛的重视.

### 10.3.6 另一个观点

多项式*预处理矩阵*的讨论指出了经典迭代法与预处理共轭梯度法之间的一个重要关系. 很多迭代法具有形式:

$$x_k=x_{k-2}+w_k(\gamma_k z_{k-1}+x_{k-1}-x_{x-2}), \tag{10.3.10}$$

其中 $\boldsymbol{M}z_{k-1}=r_{k-1}=b-\boldsymbol{A}x_{k-1}$. 例如, 若令 $w_k=1$ 和 $\gamma_k=1$, 则

$$x_k=\boldsymbol{M}^{-1}(b-\boldsymbol{A}x_{k-1})+x_{k-1},$$

即 $\boldsymbol{M}x_k=\boldsymbol{N}x_{k-1}+b$, 其中 $\boldsymbol{A}=\boldsymbol{M}-\boldsymbol{N}$. 于是, Jacobi, Gauss-Seidel, SOR, SSOR 等方法都具有形式 (10.3.10). Chebyshev 半迭代方法 (10.1.12) 也是如此.

根据 Concus, Golub, and O'Leary(1976), 可以利用形如 (10.3.10) 的主要步来组织算法 10.3.1:

$x_{-1}=0;\quad x_0=$ 初值$;\quad k=0;\quad r_0=b-Ax_0.$

**while** $r_k\neq 0$

  $k=k+1$

  解 $Mz_{k-1}=r_{k-1}$ 得 $z_{k-1}$.

  $\gamma_{k-1}=z_{k-1}^{\mathrm{T}}Mz_{k-1}/z_{k-1}^{\mathrm{T}}Az_{k-1}$

  **if** $k=1$

    $w_1=1$

  **else**

$$w_k=\left(1-\frac{\gamma_{k-1}}{\gamma_{k-2}}\frac{z_{k-1}^{\mathrm{T}}Mz_{k-1}}{z_{k-2}^{\mathrm{T}}Mz_{k-2}}\frac{1}{w_{k-1}}\right)^{-1} \tag{10.3.11}$$

  **end**

$$x_k = x_{k-2} + w_k(\gamma_{k-1}z_{k-1} + x_{k-1} - x_{k-2})$$
$$r_k = b - Ax_k$$

**end**

$x = x_n$

因此, 可以把 (10.3.11) 中的数 $w_k$ 和 $\gamma_k$ 看成是加速迭代 $\boldsymbol{Mx}_k = \boldsymbol{Nx}_{k-1} + \boldsymbol{b}$ 收敛的加速参数. 所以, 任何基于分裂 $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$ 的迭代法都可用共轭梯度法来加速, 只要 $\boldsymbol{M}$(预处理矩阵) 是对称正定的.

## 习 题

**10.3.1** 详细讨论基于 gaxpy 的 Cholesky(即算法 4.2.1) 的不完全分解算法.

**10.3.2** 在算法 10.3.1 的实际实现中, 需要多少个 $n$ 维向量的存储量? 可忽略解方程 $\boldsymbol{Mz} = \boldsymbol{r}$ 所需的工作空间.

## 本节注释与参考文献

我们关于预处理共轭梯度法的讨论来源于不同文献, 包括:

P. Concus, G. H. Golub, and D. P. O'Leary (1976). "A Generalized Conjugate Gradient Method for the Numerical Solution of Elliptic Partial Differential Equations," in *Sparse Matrix Computations*, ed. J. R. Bunch and D. J. Rose, Academic Press, New York.

G. H. Golub and G. Meurant (1983).*Résolution Numérique des Grandes Systèmes Linéaires*, Collection de la Direction des Etudes et Recherches de l'Electricité de France, Vol. 49, Eyolles, Paris.

O. Axelsson (1985), "A Survey of Preconditioned Iterative Methods for Linear Systems of Equations," *BIT* 25, 166–187

P. Concus, G. H. Golub, and G. Meurant (1985). "Block Preconditioning for the Conjugate Gradient Method," *SIAM J. Sci. and Stat. Comp.* 6, 220–252.

O.Axelsson and G. Lindskog (1986). "On the Rate of Convergence of the Preconditioned Conjugate Gradient Method," *Numer. Math.* 48, 499–523.

不完全分解思想的详细讨论见:

J. A. Meijerink and H. A. Van der vorst (1977). "An Iterative Solution Method for Linear Equation Systems of Which the Coefficient Matrix is a Symmetric *M*-Matrix," *Math. Comp.* 31, 148–162.

T. A. Mantueffel (1979). "Shifted Incomplete Cholesky Factorization," in *Sparse Matrix Proceedings*, 1978, ed I. S. Duff and G. W. Stewart, SIAM Publications, Philadelphia, PA.

T. F. Chan, K. R. Jackson, and B. Zhu (1983). "Alternating Direction Incomplete Factorizations," *SIAM J. Numer. Anal.* 20, 239–257.

G. Roderigue and D. Wolitzer (1984). "Preconditioning by Incomplete Block Cyclic Reduction," *Math. Comp.* 42, 549–566.

O. Axelsson (1985). "Incomplete Block Matrix Factorization Preconditioning Methods. The Ultimate Answer?", *J. Comput. Appl. Math.* 12&13, 3–18.

O. Axelsson (1986). "A General Incomplete Block Matrix Factorization Method," *Lin. Alg. Appl.* 74, 179–190.

H. Elman (1986). "A Stability Analysis of Incomplete LU Factorization," *Math. Comp.* 47, 191–218.

T. Chan (1991). "Fourier Analysis of Relaxed Incomplete Factorization Preconditioners," *SIAM J. Sci. Statist. Comput.* 12, 668–680.

Y. Notay (1992). "On the Robustness of Modified Incomplete Factorization Methods," *J. Comput. Math.* 40, 121–141.

关于区域分解法以及其他"PDE 驱动"的预处理思想可见:

J. H. Bramble, J. E. Pasciak, and A. H. Schatz (1986). "The construction of Preconditioners for Elliptic Problems by Substructuring I," *Math. Comp.* 47, 103–134.

J. H. Bramble, J. E. Pasciak, and A. H. Schatz (1986). "The construction of Preconditioners for Elliptic Problems by Substructuring II," *Math. Comp.* 49, 1–17.

G. Meurant (1989). "Domain Decomposition Methods for Partial Differential Equations on Parallel Computers," to appear *Int'l J. Supercomputing Applications.*

W. D. Gropp and D. E. Keyes (1992). "Domain Decomposition with Local Mesh Refinement," *SIAM J. Sci. Statist. Comput.* 13, 967–993.

D. E. Keyes, T. F. Chan, G. Meurant, J. S. Scroggs, and R. G. Voigt (eds) (1992).*Domain Decomposition Methods for Partial Differential Equations*, SIAM Publications, Phiadelphia, PA.

M. Mu. (1995). "A New family of Preconditioners for Domain Decomposition," *SIAM J. Sci. Comp.* 16, 289–306.

多项式预处理矩阵的各个方面讨论可见:

O. G. Johnson, C. A. Micchelli, and G. Paul (1983). "Polynomial Preconditioners for Conjugate Gradient Calculations," *SIAM J. Numer. Anal.* 20, 362–376.

S. C. Eisenstat (1984). "Efficient Implementation of a Class of Preconditioned Conjugate Gradient Methods," *SIAM J. Sci. and Stat. Computing.* 2, 1–4.

Y. Saad (1985). "Practical Use of Polynomial Preconditionings for the Conjugate Gradient Method," *SIAM J. Sci. and Stat. Comp.* 6, 865–882.

L. Adams (1985). "m-step Preconditioned Congugate Gradient Methods," *SIAM J. Sci. and Stat. Comp.* 6, 452–463.

S. F. Ashby (1987). "Polynomial Preconditioning for Conjugate Gradient Methods," Ph. D. Thesis, Dept. of Computer Science, University of Illinois.

S. Ashby, T. Manteuffel, and P. Saylor (1989). "Adaptive Polynomial Preconditioning for Hermitian Indefinite Linear Systems," *BIT* 29, 583-609.

R. W. Freund (1990). "On Conjugate Gradient Type Methods and Polynomial Preconditioners for a Class of Complex Non-Hermitian Matrices," *Numer. Math.* 57, 285–312.

S. Ashby, T. Manteuffel, and J. Otto (1992). "A Comparison of Adaptive Chebyshev and Least Squares Polynomial Preconditioning for Hermitian Positive Definite Linear Systems," *SIAM J. Sci. Stat. Comp.* 13, 1–29.

共轭梯度法的许多向量实现或并行实现已发展起来, 可见:

P. F. Dubois, A. Greenbaum, and G. H. Rodrigue (1979). "Approximating the Inverse of a Matrix for Use on Iterative Algorithms on Vector Processors," *Computing* 22, 257–268.

H. A. Van der Vorst (1982). "A Vectorizable Variant of Some ICCG Methods," *SIAM. J. Sci. and Stat. Comp.* 3, 350–356.

G. Meurant (1984). "The Block Preconditioned Conjugate Gradient Method on Vector Computers," *BIT* 24, 623–633.

T. Jordan (1984). "Conjugate Gradient Preconditioners for Vector and Parallel Processors," in G. Birkoff and A. Schoenstadt (eds), *Proceedings of the Conference on Elliptic Problem Solvers*, Academic Press, NY.

H. A. Van der Vorst (1986). "The Performance of Fortran Implementations for Preconditioned Conjugate Gradients on Vector Computers," *Parallel Computing* 3, 49–58.

M. K. Seager (1986). "Parallelizing Conjugate Gradient for the Cray X-MP," *Parallel Computing* 3, 35–47.

O. Axelsson and B. Polman (1986). "On Approximate Factorization Methods for Block Matrices Suitable for Vector and Parallel Processors," *Lin. Alg. and Its Applic.* 77, 3–26

D. P. O'Leary (1987). "Parallel Implementation of the Block Conjugate Gradient Algorithm," *Parallel Computers* 5, 127–140.

R. Melhem (1987). "Toward Efficient Implementation of Preconditioned Conjugate Gradient Methods on Vector Supercomputers," *Int'l J. Supercomputing Applications* 1, 70–98.

E. L. Poole and J. M. Ortega (1987). "Multicolor ICCG Methods for Vector Computers," *SIAM J. Numer. Anal.* 24, 1394–1418.

C. C. Ashcraft and R. Grimes (1988). "On Vectorizing Incomplete Factorization and SSOR Preconditioners," *SIAM J. Sci. and Stat. Comp.* 9, 122–151.

U. Meier and A. Sameh (1988). "The Behavior of Conjugate Gradient Agorithms on a Multivector Processor with a Hierarchical Memory," *J. Comput. Appl. Math.* 24, 13–32.

W. D. Gropp and D. E. Keyes (1988). "Complexity of Parallel Implementation of Domain Decomposition Techniques for Elliptic Partial Differential Equations," *SIAM J. Sci. and Stat. Comp.* 9, 312–326.

H. Van Der Vorst (1989). "High Performance Preconditioning," *SIAM J. Sci. and Stat. Comp.* 10, 1174–1185.

H. Elman (1989). "Approximate Schur Complement Preconditioners on Serial and Parallel Computers," *SIAM J. Sci. Stat. Comput.* 10, 581–605.

O. Axelsson and V. Eijkhout (1989). "Vectorizable Preconditioners for Elliptic Difference Equations in Three Space Dimensions," *J. Comput. Appl. Math.* 27, 299–321.

S. L. Johnsson and K. Mathur (1989). "Experience with the Conjugate Gradient Method for Stress Analysis on a Data Parallel Supercomputer," *International Journal on Numerical Methods in Engineering* 27, 523–546.

L. Mansfield (1991). "Damped Jacobi Preconditioning and Coarse Grid Deflation for Conjugate Gradient Iteration on Parallel Computers," *SIAM J. Sci. and Stat.Comp.* 12, 1314–1323.

V. Eijkhout (1991). "Analysis of Parallel Incomplete Point Factorizations," *Lin. Alg. and Its Applic.* 154–156, 723–740.

S. Doi (1991). "On Parallelism and Convergence of Incomplete LU Factorizations," *Appl. Numer. Math.* 7, 417–436.

下文讨论了大型 Toeplitz 方程组的预处理矩阵:

G. Strang (1986). "A Proposal for Toeplitz Matrix Calculations," *Stud. Appl. Math.* 74, 171–176.

T. F. Chan (1988). "An Optimal Circulant Preconditioner for Toeplitz Systems," *SIAM J. Sci. Stat. Comp.* 9, 766–771.

R. H. Chan (1989). "The Spectrum of a Family of Circulant Preconditioned Toeplitz Systems," *SIAM J. Num. Anal.* 26, 503–506.

R. H. Chan (1991), "Preconditioners for Toeplitz Systems with Nonnegative Generating Functions" *IMA J. Num. Anal,* 11, 333–345.

T. Huckle (1992). "Circulant and Skewcirculant Matrices for Solving Toeplitz Matrix Problems," *SIAM J. Matrix Anal. Appl.* 13, 767–777.

T. Huckle (1992). "A Note on Skew-Circulant Preconditioners for Elliptic Problems," *Numerical Algorithms* 2, 279–286.

R. H. Chan, J. G. Nagy, and R. J. Plemmons (1993). "FFT based Preconditioners for Toeplitz Block Least Squares Problems," *SIAM J. Num. Anal.* 30, 1740–1768.

M. Hanke and J. G. Nagy (1994). "Toeplitz Approximate Inverse Preconditioner for Banded Toeplitz Matrices," *Numerical Algorithms* 7, 183–199.

R. H. Chan, J. G. Nagy, and R. J. Plemmons (1994). "Circulant Preconditioned Toeplitz Least Squares Iterations," *SIAM J. Matrix Anal. Appl.* 15, 80–97.

T. F. Chan and J. A. Olkin (1994). "Circulant Preconditioners for Toeplitz Block Matrices," *Numerical Algorithms* 6, 89–101.

最后, 我们给一些关于共轭梯度法实际用途的文献:

J. K. Reid (1972). "The Use of Conjugate Gradients for Systems of Linear Equations Possessing Property A," *SIAM J. Num. Anal.* 9, 325–332.

D. P. O'Leary (1980). "The Block Conjugate Gradient Algorithm and Related Methods," *Lin. Alg. and Its Applic.* 29, 293–322.

R. C. Chin, T. A. Manteuffel, and J. de Pillis (1984). "ADI as a Preconditioning for Solving the Convection-Diffusion Equation," *SIAM J. Sci. and Stat. Comp.* 5, 281–299.

I. Duff and G. Meurant (1989). "The Effect of Ordering on Preconditioned Conjugate Gradients," *BIT* 29, 635–657.

A. Greenbaum and G. Rodrigue (1989). "Optimal Preconditioners of a Given Sparsity Pattern," *BIT* 29, 610–634.

O. Axelsson and P. Vassilevski (1989). "Algebraic Multilevel Preconditioning Methods I," *Numer. Math.* 56, 157–177.

O. Axelsson and P. Vassilevski (1990). "Algebraic Multilevel Preconditioning Methods II," *SIAM J. Numer. Anal.* 27, 1569–1590.

M. Hanke and M. Neumann (1990). "Preconditionings and Splittings for Rectangular Systems," *Numer. Math.* 57, 85–96.

A. Greenbaum (1992). "Diagonal Scalings of the Laplacian as preconditioners for Other Elliptic Differential Operators," *SIAM J. Matrix Anal. Appl.* 13, 826–846.

P. E. Gill, W. Murray, D. B. Poncele6n, and M. A. Saunders (1992). "Preconditioners for Indefinite Systems Arising in Optimization," *SIAM J. Matrix Anal. Appl.* 13, 292–311.

G. Meurant (1992). "A Review on the Inverse of Symmetric Tridiagonal and Block Tridiagonal Matrices," *SIAM J. Matrix Anal. Appl.* 13, 707–728.

S. Holmgren and K. Otto (1992). "Iterative Solution Methods and Preconditioners for Block-Tridiagonal Systems of Equations," *SIAM J. Matrix Anal. Appl.* 13, 863–886.

S. A. Vavasis (1992). "Preconditioning for Boundary Integral Equations," *SIAM J. Matrix Anal. Appl.* 13, 905–925.

P. Joly and G. Meurant (1993). "Complex Conjugate Gradient Methods," *Numerical Algorithms* 4, 379–406.

X. -C. Cai and O. Widlund (1993). "Multiplicative Schwarz Algorithms for Some Nonsymmetric and Indefinite Problems," *SIAM J. Numer. Anal.* 30, 936–952.

## 10.4 其他 Krylov 子空间方法

前两节介绍的共轭梯度法适用于对称正定方程组. 9.3.2 节中介绍的对称 Lanczos 方法的变化形式 MINRES 和 SYMMLQ 方法能处理对称非正定问题. 为了得到可适用于非对称问题的迭代法, 我们还要作进一步的推广.

这里我们仿效 Freund, Golub, and Nachtigal(1992) 的综述文章及 Golub and Ortega (1993) 的第 9 章来进行论述. 重点是涉及 Krylov 空间上优化的共轭梯度型算法.

应该记住, 我们的算法描述和软件产品之间有很大差距. 关于这一点, Barrett *et al* (1993) 的"样板"书作了很好的描述. Saad(1996) 的著作也是很值得推荐的.

### 10.4.1 法方程方法

解最小二乘问题的法方程方法是很吸引人的, 因为它可以利用简单的 Cholesky 分解技巧而不需要复杂的正交化方法. 同样, 对非对称问题 $\boldsymbol{Ax}=\boldsymbol{b}$, 用已有的共轭梯度法解等价的对称正定方程组

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Ax}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$$

也是有诱惑力的. 实际上, 如果在算法 10.2.1 中用 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 代替 $\boldsymbol{A}$, 并注意到法方程中的残量 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Ax}_k$ 是"真实"残量 $\boldsymbol{b}-\boldsymbol{Ax}_k$ 的 $\boldsymbol{A}^{\mathrm{T}}$ 倍, 我们就可以得到下面的共轭梯度法方程残量法 (CGNR).

**算法 10.4.1(CGNR)** *若矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 非奇异, $\boldsymbol{b}\in\mathbb{R}^n, \boldsymbol{x}_0\in\mathbb{R}^n$ 为初值, 则本算法可计算出方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解 $\boldsymbol{x}\in\mathbb{R}^n$:*

$k=0$
$r_0=b-Ax_0$
**while** $r_k\neq 0$
  $k=k+1$
  **if** $k=1$
    $p_1=A^{\mathrm{T}}r_0$
  **else**
    $\beta_k=(A^{\mathrm{T}}r_{k-1})^{\mathrm{T}}(A^{\mathrm{T}}r_{k-1})/(A^{\mathrm{T}}r_{k-2})^{\mathrm{T}}(A^{\mathrm{T}}r_{k-2})$
    $p_k=A^{\mathrm{T}}r_{k-1}+\beta_k p_{k-1}$
  **end**
  $\alpha_k=(A^{\mathrm{T}}r_{k-1})^{\mathrm{T}}(A^{\mathrm{T}}r_{k-1})/(Ap_k)^{\mathrm{T}}(Ap_k)$
  $x_k=x_{k-1}+\alpha_k p_k$
  $r_k=r_{k-1}-\alpha_k Ap_k$
**end**
$x=x_k$

另一种使非对称问题 $\boldsymbol{Ax}=\boldsymbol{b}$“共轭梯度友好”的方式是解如下方程组:

$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}=\boldsymbol{b},\quad \boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}.$$

在“$\boldsymbol{y}$ 空间”中，共轭梯度算法有如下形式:

$k=0$
$y_0=$初值$(AA^{\mathrm{T}}y_0=b)$
$r_0=b-AA^{\mathrm{T}}y_0$
**while** $r_k\neq 0$
　　$k=k+1$
　　**if** $k=1$
　　　　$p_1=r_0$
　　**else**
　　　　$\beta_k=r_{k-1}^{\mathrm{T}}r_{k-1}/r_{k-2}^{\mathrm{T}}r_{k-2}$
　　　　$p_k=r_{k-1}+\beta_k p_{k-1}$
　　**end**
　　$\alpha_k=r_{k-1}^{\mathrm{T}}r_{k-1}/p_k^{\mathrm{T}}AA^{\mathrm{T}}p_k$
　　$y_k=y_{k-1}+\alpha_k p_k$
　　$r_k=r_{k-1}-\alpha_k AA^{\mathrm{T}}p_k$
**end**
$y=y_k$

作变量替换 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}_k\to\boldsymbol{x}_k,\boldsymbol{A}^{\mathrm{T}}\boldsymbol{p}_k\to\boldsymbol{p}_k$, 再作简化则得到共轭梯度法方程误差法, 简称 CGNE.

**算法 10.4.2(CGNE)**　若矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 非奇异, $\boldsymbol{b}\in\mathbb{R}^n,\boldsymbol{x}_0\in\mathbb{R}^n$ 为初值 $(\boldsymbol{Ax}_0\approx\boldsymbol{b})$, 则本算法计算出 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解 $\boldsymbol{x}\in\mathbb{R}^n$:

$k=0$
$r_0=b-Ax_0$
**while** $r_k\neq 0$
　　$k=k+1$
　　**if** $k=1$
　　　　$p_1=A^{\mathrm{T}}r_0$
　　**else**
　　　　$\beta_k=r_{k-1}^{\mathrm{T}}r_{k-1}/r_{k-2}^{\mathrm{T}}r_{k-2}$
　　　　$p_k=A^{\mathrm{T}}r_{k-1}+\beta_k p_{k-1}$
　　**end**
　　$\alpha_k=r_{k-1}^{\mathrm{T}}r_{k-1}/p_k^{\mathrm{T}}p_k$
　　$x_k=x_{k-1}+\alpha_k p_k$
　　$r_k=r_{k-1}-\alpha_k Ap_k$

**end**

$x = x_k$

一般来讲, 这两种法方程方法的困难在于矩阵的条件数变成了原来的平方, (回忆一下定理 10.2.6). 然而, 在有些情况下他们是很有效的, 这方面的论述可见 Freund, Golub and Nachtigal (1991).

### 10.4.2 目标函数的注记

基于我们关于共轭梯度法的知识, CGNR 迭代所产生的 $\boldsymbol{x}_k$ 使得函数

$$\phi_1(x) = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{x} - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$$

在集合

$$S_k^{(\mathrm{CGNR})} = \boldsymbol{x}_0 + K(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{r}_0, k)$$

上达到极小. 容易证明

$$\frac{1}{2} \parallel \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \parallel_2^2 = \phi_1(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{b},$$

所以 $\boldsymbol{x}_k$ 使得残量在 $S_k^{(\mathrm{CGNR})}$ 上达到极小. “CGNR” 中的 R 表示对残量 (residual) 是最优的.

另一方面 CGNE(隐含) 迭代产生的 $\boldsymbol{y}_k$ 使函数

$$\phi_2(\boldsymbol{y}) = \frac{1}{2}\boldsymbol{y}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})\boldsymbol{y} - \boldsymbol{y}^{\mathrm{T}}\boldsymbol{b}$$

在集合 $y_0 = K(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{b} - \boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}_0, k)$ 上达到极小. 作变量替换 $\boldsymbol{x} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}$, 可证明 $x_k$ 使得

$$\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x} - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b} = \frac{1}{2} \parallel \boldsymbol{x} - \boldsymbol{A}^{-1}\boldsymbol{b} \parallel_2^2 + \frac{1}{2} \parallel \boldsymbol{A}^{-1}\boldsymbol{b} \parallel_2^2$$

在集合

$$S_k^{(\mathrm{CGNE})} = \boldsymbol{x}_0 + K(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\boldsymbol{r}_0, k) \tag{10.4.1}$$

上达到最小. 因此在 CGNE 方法中, 第一步都使误差达到极小, 这就是“CGNE”中“E”的含义.

### 10.4.3 共轭残量法

我们记得, 当 $\boldsymbol{A}$ 是对称正定矩阵时, 它有一对称正定的平方根 $\boldsymbol{A}^{1/2}$(见 4.2.10 节). 在这种情况下, 方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 和 $\boldsymbol{A}^{1/2}\boldsymbol{x} = \boldsymbol{A}^{-1/2}\boldsymbol{b}$ 相互等价. 前者是后者的法方程形式. 如果我们对这个平方根方程用 CGNR 方法, 并对结果作一下简化, 则得如下算法.

**算法 10.4.3(共轭残量法)** 如果矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 对称正定, $\boldsymbol{b} \in \mathbb{R}^n, \boldsymbol{x}_0 \in \mathbb{R}^n$ 为初值 $(\boldsymbol{A}\boldsymbol{x}^0 \approx \boldsymbol{b})$, 则本算法可计算 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的解.

$k = 0$

$r_0 = b - Ax_0$

**while** $r_k \neq 0$
　　$k = k+1$
　　**if** $k = 1$
　　　　$p_1 = r_0$
　　**else**
　　　　$\beta_k = r_{k-1}^{\mathrm{T}} A r_{k-1} / r_{k-2}^{\mathrm{T}} A r_{k-2}$
　　　　$Ap_k = Ar_{k-1} + \beta_k Ap_{k-1}$
　　**end**
　　$\alpha_k = r_{k-1}^{\mathrm{T}} A r_{k-1} / (Ap_k)^{\mathrm{T}}(Ap_k)$
　　$x_k = x_{k-1} + \alpha_k p_k$
　　$r_k = r_{k-1} - \alpha_k Ap_k$
**end**
$x = x_k$

从对 CGNR 方法的评价中可得, 在第 $k$ 步迭代, $\| \boldsymbol{A}^{-1/2}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}) \|_2$ 在集合 $\boldsymbol{x}_0 + K(\boldsymbol{A}, \boldsymbol{r}_0, k)$ 上达到极小.

### 10.4.4 GMRES 方法

在 9.3.2 节中, 针对对称非正定问题 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, 我们简要讨论了基于 Lanczos 的 MINRES 方法. 在该算法中, 第 $k$ 步迭代点 $\boldsymbol{x}_k$ 使得 $\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x} \|_2$ 在集合

$$S_k = x_0 + \operatorname{span}\{r_0, \boldsymbol{A}r_0, \cdots, \boldsymbol{A}^{k-1}r_0\} = x_0 + K(\boldsymbol{A}, r_0, k) \tag{10.4.2}$$

上达到极小. 算法的关键思想是, 用 Lanczos 向量 $q_1, q_2, \cdots, q_k$ 来表示 $x_k$. 当 $q_1$ 是初始残量 $r_0 = b - \boldsymbol{A}x_0$ 的倍数时, $q_1, \cdots, q_k$ 可生成 $\boldsymbol{K}(\boldsymbol{A}, r_0, k)$.

在 Saad and Schultz (1986) 提出的广义极小残量(GMRES) 方法中, 采有了同一途径, 不同之处在于迭代点用 Arnoldi 向量而不是 Lanczos 向量来表示, 其目的是为了能处理非对称矩阵 $\boldsymbol{A}$. Arnoldi 迭代 (9.4.1) 经过 $k$ 步以后, 产生如下分解:

$$\boldsymbol{A}\boldsymbol{Q}_k = \boldsymbol{Q}_{k+1}\tilde{\boldsymbol{H}}_k, \tag{10.4.3}$$

其中 $\boldsymbol{Q}_{k+1} = [\boldsymbol{Q}_k \;\; q_{k+1}]$ 的列是标准正交 Arnoldi 向量,

$$\tilde{\boldsymbol{H}}_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2k} \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & h_{k,k-1} & h_{k,k} \\ 0 & \cdots & \cdots & 0 & h_{k+1,k} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$$

为上 Hessenberg 矩阵. 在 GMRES 方法的第 $k$ 步, 在约束条件 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{Q}_k \boldsymbol{y}_k$ ($\boldsymbol{y}_k \in \mathbb{R}^k$) 之下, $\| \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k \|_2$ 达到极小. 若 $q_1 = \boldsymbol{r}_0/\rho_0$, $\rho_0 = \| \boldsymbol{r}_0 \|_2$, 则得到:

$$\begin{aligned}\| \boldsymbol{b}-\boldsymbol{A}(\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y}_k) \|_2 &= \| \boldsymbol{r}_0-\boldsymbol{A}\boldsymbol{Q}_k\boldsymbol{y}_k \|_2\\ &= \| \boldsymbol{r}_0-\boldsymbol{Q}_{k+1}\tilde{\boldsymbol{H}}_k\boldsymbol{y}_k \|_2\\ &= \| \rho_0\boldsymbol{e}_1-\tilde{\boldsymbol{H}}_k\boldsymbol{y}_k \|_2 .\end{aligned}$$

因此, $\boldsymbol{y}_k$ 是一个 $(k+1)\times k$ 的最小二乘问题的解, GMRES 迭代为 $\boldsymbol{x}_k=\boldsymbol{x}_0+\boldsymbol{Q}_k\boldsymbol{y}_k$.

**算法 10.4.4(GMRES)** 如果 $\boldsymbol{A}\in\mathbb{R}^{n\times n}$ 非奇异, $\boldsymbol{b}\in\mathbb{R}^n,\boldsymbol{x}_0\in\mathbb{R}^n$ 为初始向量 $(\boldsymbol{A}\boldsymbol{x}_0\approx\boldsymbol{b})$, 则本算法计算出 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}$ 的解 $x\in\mathbb{R}^n$:

$r_0=b-Ax_0$

$h_{10}=\| r_0 \|_2$

$k=0$

**while** $h_{k+1,k}>0$

  $q_{k+1}=r_k/h_{k+1,k}$

  $k=k+1$

  $r_k=Aq_k$

  **for** $i=1:k$

    $h_{ik}=q_i^{\mathrm{T}}r_k$

    $r_k=r_k-h_{ik}q_i$

  **end**

  $h_{k+1,k}=\| r_k \|_2$

  $x_k=x_0+\boldsymbol{Q}_k y_k$, 其中 $\| h_{10}e_1-\tilde{\boldsymbol{H}}_k y_k \|_2=\min$

**end**

$x=x_k$

容易证明 $\| b-\boldsymbol{A}x_k \|_2=h_{k+1,k}$. 上 Hessenberg 矩阵的最小二问题能用 Givens 旋转有效求解. 在实际计算中, 只有当残量达到我们所希望的范围时, 才需要形成 $x_k$.

“无界的 GMRES ”的主要问题是: 在第 $k$ 步迭代中, 需 $O(kn)$ 个 flop . 因此, 与 Arnoldi 方法一样, 一个实际的 GMRES 实现需要采用再启动技巧, 以避免过多的计算量和存储量. 例如, 若最多容许计算 $m$ 步, 则 $x_m$ 可作为下一个 GMRES 序列的初始向量.

### 10.4.5 预处理

预处理是使 GMRES 方法有效的另一关键因素. 与 10.3 节中的预处理共轭梯度法类似, 我们得到一非奇异矩阵 $\boldsymbol{M}=\boldsymbol{M}_1\boldsymbol{M}_2$, 它在某种意义下逼近矩阵 $\boldsymbol{A}$. 然后对方程组 $\tilde{\boldsymbol{A}}\,\tilde{\boldsymbol{x}}=\tilde{\boldsymbol{b}}$ 用 GMRES 方法, 其中 $\tilde{\boldsymbol{A}}=\boldsymbol{M}_1^{-1}\boldsymbol{A}\boldsymbol{M}_2^{-1},\tilde{\boldsymbol{b}}=\boldsymbol{M}_1^{-1}\boldsymbol{b},\tilde{\boldsymbol{x}}=\boldsymbol{M}_2\boldsymbol{x}$. 如果我们对这个波形号方程组写出相应的 GMRES 迭代算法, 并整理方程使得只存储原始变量, 则所得到的迭代中需要解预处理矩阵为 $\boldsymbol{M}$ 的线性方程组. 所以, 寻找一个好的预处理矩阵 $\boldsymbol{M}=\boldsymbol{M}_1\boldsymbol{M}_2$ 就是在以 $\boldsymbol{M}$ 为系数矩阵的方程容易求解的前提下, 使得 $\tilde{\boldsymbol{A}}=\boldsymbol{M}_1^{-1}\boldsymbol{A}\boldsymbol{M}_2^{-1}$ 尽量“像”单位矩阵.

### 10.4.6 双共轭梯度法

就像 Arnoldi 方法推广出 GMRES 方法一样, 非对称的 Lanczos 方法可推广出双共轭梯度法 (BiCG). BiCG 的出发点可追溯到 9.3.1 节中由 Lanczos 算法导出共轭梯度法. 利用 Lanczos 向量, 共轭梯度法的第 $k$ 次迭代可表示为 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{Q}_k\boldsymbol{y}_k$, 其中 $\boldsymbol{Q}_k$ 为 Lanczos 向量组成的矩阵, $\boldsymbol{T}_k = \boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k$ 为三对角矩阵, $\boldsymbol{y}_k$ 为方程组 $\boldsymbol{T}_k\boldsymbol{y}_k = \boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{r}_0$ 的解. 注意关系式:

$$\boldsymbol{Q}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k) = \boldsymbol{Q}_k^{\mathrm{T}}(\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{Q}_k\boldsymbol{y}_k) = 0.$$

所以, 我们可以通过要求 $\boldsymbol{x}_k$ 为集合 $\boldsymbol{x}_0 + \boldsymbol{K}(\boldsymbol{A}, \boldsymbol{r}_0, k)$ 的元素并且它产生一个与子空间 $\boldsymbol{K}(\boldsymbol{A}, \boldsymbol{r}_0, k)$ 正交的残量来刻画 $x_k$.

在矩阵为非对称的情形中, 我们可以推广这一想法: 产生迭代序列 $\{\boldsymbol{x}_k\}, \boldsymbol{x}_k \in \boldsymbol{x}_0 + \boldsymbol{K}(\boldsymbol{A}, \boldsymbol{r}_0, k)$, 相应的残量与 $\boldsymbol{K}(\boldsymbol{A}^{\mathrm{T}}, \boldsymbol{s}_0, k)$ 正交, $\boldsymbol{s}_0 \in \mathbb{R}^n$. 如果用非对称 Lanczos 方法来产生两个 Krylov 空间的基, 则算法可得到简化. 特别地, 非对称 Lanczos 算法 (9.4.7) 经过 $k$ 步以后, 可得到 $\boldsymbol{Q}_k, \boldsymbol{P}_k \in \mathbb{R}^{n\times k}, \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{Q}_k = \boldsymbol{I}_k$ 和三对角矩阵 $\boldsymbol{T}_k = \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}_k$, 使得

$$\begin{aligned} \boldsymbol{A}\boldsymbol{Q}_k &= \boldsymbol{Q}_k\boldsymbol{T}_k + r_k\boldsymbol{e}_k^{\mathrm{T}}, \quad \boldsymbol{P}_k^{\mathrm{T}}\boldsymbol{r}_k = \boldsymbol{0}; \\ \boldsymbol{A}^{\mathrm{T}}\boldsymbol{P}_k &= \boldsymbol{P}_k\boldsymbol{T}_k^{\mathrm{T}} + s_k\boldsymbol{e}_k^{\mathrm{T}}, \quad \boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{s}_k = \boldsymbol{0}. \end{aligned} \tag{10.4.4}$$

在 BiCG 中, 令 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{Q}_k\boldsymbol{y}_k$, 其中 $\boldsymbol{T}_k\boldsymbol{y}_k = \boldsymbol{Q}_k^{\mathrm{T}}\boldsymbol{r}_0$. 注意, Galerkin 条件$\boldsymbol{P}_k^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k) = \boldsymbol{P}_k^{\mathrm{T}}(\boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{Q}_k\boldsymbol{y}_k) = \boldsymbol{0}$ 成立.

像所期望的那样, 可以找出从 $x_{k-1}$ 和 $q_{k-1}$ 的简单组合计算出 $x_k$ 的迭代公式, 而不是用前面所有的 $q$ 向量的线性组合.

BiCG 方法受到 "严重失败" 的制约, 因为它依赖于非对称 Lanczos 方法. 然而, 采用向前看的 Lanczos 方法, 有可能克服这方面的一些困难.

### 10.4.7 QMR 方法

从非对称 Lanczos 方法导出的另一种迭代法是 Freund and Nachtigal (1991) 提出的拟极小残量法 (QMR). 和 BiCG 一样, 第 $k$ 步迭代同样具有形式 $\boldsymbol{x}_k = \boldsymbol{x}_0 + \boldsymbol{Q}_k\boldsymbol{y}_k$. 易证, (9.4.7) 经过 $k$ 步后有分解:

$$\boldsymbol{A}\boldsymbol{Q}_k = \boldsymbol{Q}_{k+1}\widetilde{\boldsymbol{T}}_k,$$

其中 $\tilde{\boldsymbol{T}}_k \in \mathbb{R}^{(k+1)\times k}$ 是三对角的. 由此可知, 若 $\boldsymbol{q}_1 = \rho(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$, 则有

$$\begin{aligned} \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k &= \boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_0 + \boldsymbol{Q}_k\boldsymbol{y}_k) = \boldsymbol{r}_0 - \boldsymbol{A}\boldsymbol{Q}_k\boldsymbol{y}_k \\ &= \boldsymbol{r}_0 - \boldsymbol{Q}_{k+1}\widetilde{\boldsymbol{T}}_k\boldsymbol{y}_k = \boldsymbol{Q}_{k+1}(\rho e_1 - \widetilde{\boldsymbol{T}}_k\boldsymbol{y}_k). \end{aligned}$$

如果选择 $\boldsymbol{y}_k$ 使得这个向量的 2 范数极小, 则在精确运算下, $x_0 + \boldsymbol{Q}_k\boldsymbol{y}_k$ 正好是 GMRES 迭代所产生的点. 在 QMR 中, 令 $y_k$ 使得 $\| \rho e_1 - \tilde{\boldsymbol{T}}_k\boldsymbol{y}_k \|_2$ 极小.

### 10.4.8 总结

我们所介绍的方法并没有绝对的好坏之分. 方法的选择是复杂的, 它依赖于很多因素. Barrett *et al* (1993) 对主要算法给出了特别有说服力的评价.

## 习　题

**10.4.1** 类似于 (10.2.16), 导出 CGNR 法、CGNE 法和共轭残量法的有效实现方法.

**10.4.2** 建立 CGNR 法与 9.3.4 节中概括的 LSQR 法的数学等价关系.

**10.4.3** 证明 (10.4.3)

**10.4.4** 导出预处理 GMRES 的实现, 参照 10.3 节中预处理共轭梯度法的处理方式. [特别注意 (10.3.2) 和 (10.3.3)].

**10.4.5** 证明 GMRES 最小二乘问题满秩.

## 本节注释与参考文献

下面的文献对非对称迭代法提供了很好的入门:

S. Eisenstat, H. Elman, and M. Schultz (1983). "Variational Iterative Methods for Nonsymmetric Systems of Equations," *SIAM J. Num. Anal.* 20, 345–357.

R. W. Freund, G. H. Golub, and N. Nachtigal (1992). "Iterative Solution of Linear Systems," *Acta Numerica*, 1, 57–100.

N. Nachtigal, S. Peddy, and L. Trefethen (1992). "How Fast Are Nonsymmetric Matrix Iterations," *SIAM J. Matrix Anal. Appl.* 13, 778–795.

A. Greenbaum and L. N. Trefethen (1994). "GMRES/CR and Arnoldi/Lanczos as Matrix Approximation Problems," *SIAM J. Sci. Comp.* 15, 359–368.

Krylov 空间方法以及分析在下文中讨论:

W. E. Arnoldi (1951). "The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem," *Quart. Appl. Math.* 9, 17–29.

Y. Saad (1981). "Krylov Subspace Methods for Solving Large Unsymmetric Linear Systems," *Math. Comp.* 37, 105–126.

Y. Saad (1984). "Practical Use of Some Krylov Subspace Methods for Solving Indefinite and Nonsymmetric Linear Systems," *SIAM J. Sci. and Stat. Comp.* 5, 203–228.

Y. Saad (1989). "Krylov Subspace Methods on Supercomputers," *SIAM J. Sci. and Stat. Comp.* 10, 1200–1322.

C. -M. Huang and D. P. O'Leary (1993). "A Krylov Multisplitting Algorithm for Solving Linear Systems of Equations," *Lin. Alg. and Its Applic.* 194. 9–29.

C. C. Paige, B. N. Parlett, and H. A. Van Der Vorst (1995). "Approximate Solutions and Eigenvalue Bounds from Krylov Subspaces," *Numer, Linear Algebra with Applic.* 2, 115–134.

GMRES 方法的参数文献包括：

Y. Saad and M. Schultz (1986). "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM J. Scientific and Stat. Comp.* 7, 856–869.

H. F. Walker (1988). "Impiementation of the GMRES Method Using Householder Transformations," *SIAM J. Sci. Stat. Comp.* 9, 152–163.

C. Vuik and H. A. van der Vorst (1992). "A Comparison of Some GMRES-like Methods," *Lin. Alg. and Its Applic.* 160, 131–162.

N. Nachtigal, L. Reichel, and L. Trefethen (1992). "A Hybrid GMRES Algorithm for Nonsymmetric Linear Systems," *SIAM J. Matrix Anal. Appl.* 13, 796–825.

Y. Saad (1993). "A Flexible Inner-Outer Preconditioned GMRES Algorithm," *SIAM J. Sci. Comput.* 14, 461–469.

Z. Bai, D. Hu, and L. Reichel (1994). "A Newton Basis GMRES Implementation." *IMA J. Num. Anal.* 14, 563–581.

R. B. Morgan (1995). "A Restarted GMRES Method Augmented with Eigenvectors," *SIAM J. Matrix Anal. Applic.* 16, 1154–1171.

非对称问题的预处理思想在下文中讨论：

Y. Saad (1988). "Preconditioning Techniques for Indefinite and Nonsymmetric Linear Systems," *J. Comput. Appl. Math.* 24, 89–105.

L. Yu. Kolotilina and A. Yu. Yeremin (1993). "Factorized Sparse Approximate Inverse Preconditioning I: Theory," *SIAM J. Matrix Anal. Applic.* 14, 45–58.

I. E. Kaporin (1994). "New Convergence Results and Preconditioning Strategies for the Conjugate Gradient Method," *Num. Lin. Alg. Applic.* 1, 179–210.

L. Yu. Kolotilina and A. Yu. Yeremin (1995). "Factorized Sparse Approximate Inverse Preconditioning II: Soluion of 3D FE Systems on Massively Parallel Computers," *Intern. J. High Speed Comput.* 7, 191–215.

H. Elman (1996). "Fast Nonsymmetric Iterations and Preconditioning for Navier-Stokes Equations," *SIAM J. Sci. Comput.* 17, 33–46.

M. Benzi, C. D. Meyer, and M. Tuma (1996). "A Sparse Approximate Inverse Preconditioner for the Conjugate Gradient Method," *SIAM J. Sci. Comput.* 17, to appear.

关于发展非对称共轭梯度法的代表性文章包括：

D. M. Young and K. C. Jea (1980). "Generalized Conjugate Gradient Acceleration of Nonsymmetrizable Iterative Methods," *Lin. Alg. and Its Applic.* 34, 159–194.

O. Axelsson (1980). "Conjugate Gradient Type Methods for Unsymmetric and Inconsistent Systems of Linear Equations," *Lin Alg. and Its Applic.* 29, 1–16.

K. C. Jea and D. M. Young (1983). "On the Simplification of Generalized Conjugate Gradient Methods for Nonsymmetrizable Linear Systems," *Lin Alg. and Its Applic.* 52/53, 399–417.

V. Faber and T. Manteuffel (1984). "Necessary and Sufficient Conditions for the Existence of a Conjugate Gradient Method," *SIAM J. Numer. Anal.* 21, 352–362.

Y. Saad and M. Schultz (1985). "Conjugate Gradient-Like Algorithms for Solving Nonsymmetric Linear Systems," *Math. Comp.* 44, 417–424.

H. A. Van der Vorst (1986). "An Iterative Solution Method for Solving $f(A)x = b$ Using Krylov Subspace Information Obtained for the Symmetric Positive Definite Matrix A," *J. Comp. and App. Math.* 18, 249–263.

M. A. Saunders, H. D. Simon, and E. L. Yip (1988). "Two Conjugate Gradient-Type Methods for Unsymmetric Linear Equations," *SIAM J. Num. Anal.* 25, 927–940.

R. Freund (1992). "Conjugate Gradient-Type Methods for Linear Systems with Complex Symmetric Coefficient Matrices," *SIAM J. Sci. Statist. Comput.* 13, 425–448.

更多的基于 Lanczos 的算法在下文中讨论:

Y. Saad (1982). "The Lanczos Biorthogonalization Algorithm and Other Oblique Projection Methods for Solving Large Unsymmetric Systems," *SIAM J. Numer. Anal.* 19, 485–506.

Y. Saad (1987). "On the Lznczos Method for Solving Symmetric Systems with Several Right Hand Sides," *Math. Comp.* 48, 651–662.

C. Brezinski and H. Sadok (1991). "Avoiding Breakdown in the CGS Algorithm." *Numer, Alg.* 1, 199–206.

C. Brezinski, M. Zaglia, and H. Sadok (1992). "A Breakdown Free Lanczos Type Algorithm for Solving Linear Systems," *Numer. Math.* 63, 29–38.

S. K. Kim and A. T. Chronopoulos (1991). "A Class of Lanczos-Like Algorithms Implemented on Parallel Computers," *Parallel Comput.* 17, 763–778.

W. Joubert (1992). "Lanczos Methods for the Solution of Nonsymmetric Systems of Linear Equations," *SIAM J. Matrix Anal. Appl.* 13, 926–943.

R. W. Freund, M. Gutknecht, and N. Nachtigal (1993). "An Implementation of the Look-Ahead Lanczos Algorithm for Non-Hermitian Matrices," *SIAM J. Sci. and Stat. Comp.* 14, 137–158.

QMR 方法在下文中详细讨论:

R. W. Freund and N. Nachtigal (1991). "QMR: A Quasi-Minimal Residual Method for Non-Hermitian Linear Systems," *Numer. Math.* 60, 315–339.

R. W. Freund (1993). "A Transpose-Free Quasi-Minimun Residual Algorithm for Nonhermitian Linear System," *SIAM J. Sci. Comput.* 14, 470–482.

R. W. Freund and N. M. Nachtigal (1994). "An Implementation of the QMR Method Based on Coupled Two-term Recurrences," *SIAM J. Sci. Comp.* 15, 313–337.

双共轭梯度法的残量能揭露算法的反常表现, 从而促进稳定技术的发展:

H. van der Vorst (1992). "BiCGSTAB: A Fast and Smoothly Converging Variant of the Bi-CG for the Solution of Nonsymmetric Linear Systems," *SIAM J. Sci. and Stat. Comp.* 13, 631–644.

M. Gutknecht (1993). "Variants of BiCBSTAB for Matrices with Complex Spectrum," *SIAM J. Sci. and Stat. Comp.* 14, 1020–1033.

G. L. G. Sleijpen and D. R. Fokkema (1993). "BICGSTAB($l$) for Linear Equations Involving Unsymmetric Matrices with Complex Spectrum," *Electronic Transactions on Numerical Analysis* 1, 11–32.

C. Brezinski and M. Redivo-Zaglia (1995). "Look-Ahead in BiCGSTAB and Other Product-Type Methods for Linear Systems," *BIT* 35, 169–201.

在一些应用中, 同时对矩阵与向量相乘 $Ax$ 与 $A^{\mathrm{T}}x$ 产生了程序是很拙笨的. 不用转置的方法很流行, 可见:

P. Sonneveld (1989). "CGS, A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems," *SIAM J. Sci. and Stat. Comp.* 10, 36–52.

G. Radicati di Brozolo and Y. Robert (1989). "Parallel Conjugate Gradient-like Algorithms for Solving Sparse Nonsymmetric Linear Systems on a Vector Multiprocessor," *Parallel Computing* 11, 233–240.

C. Brezinski and M. Redivo-Zaglia (1994). "Treatment of Near-Breakdown in the CGS Algorithms," *Numerical Algorithms* 7, 33–73.

E. M. Kasenally (1995). "GMBACK: A Generalized Minimum Backward Error Algorithm for Nonsymmetric Linear Systems," *SIAM J. Sci. Comp.* 16, 698–719.

C. C. Paige, B. N. Parlett, and H. A. van der Vorst (1995). "Approximate Solutions and Eigenvalue Bounds from Krylov Subspaces," *Num. Lin. Alg. with Applic.* 2, 115-133.

M. Hochbruck and Ch. Lubich (1996), "On Krylov Subspace Approximations to the Matrix Exponential Operator," *SIAM J. Numer. Anal.* , to appear.

M. Hochbruck and Ch. Lubich (1996), "Error Analysis of Krylov Method in a Nutshell," *SIAM J.Sci. Comput.* , to appear.

关于长方阵 $A$ 的伪逆与用于 $A^{\mathrm{T}}A$ 的共轭梯度法之间的联系可见:

M. Hestenes (1975). "Pseudoinverses and Conjugate Gradients," *CACM* 18, 40–43.

# 第11章　矩阵函数

在控制论和其他应用领域,经常出现自变量为一方阵 $\boldsymbol{A}$ 的函数 $f(\boldsymbol{A})$ 的计算问题. 粗略地说, 如果 $f(z)$ 是定义在 $\lambda(\boldsymbol{A})$ 上的标量函数, 则在 $f(z)$ 的表达式中以 $\boldsymbol{A}$ 代替 $z$, 就可定义 $f(\boldsymbol{A})$. 例如, 若 $f(z)=(1+z)/(1-z)$, 且 $1\notin\lambda(\boldsymbol{A})$, 则 $f(\boldsymbol{A})=(\boldsymbol{I}+\boldsymbol{A})(\boldsymbol{I}-\boldsymbol{A})^{-1}$.

当 $f$ 为超越函数时, $f(\boldsymbol{A})$ 的计算变得非常有趣. 在这种更为复杂的情形中,一种方法是计算 $\boldsymbol{A}$ 的特征值分解 $\boldsymbol{A}=\boldsymbol{YBY}^{-1}$, 并利用公式 $f(\boldsymbol{A})=\boldsymbol{Y}f(\boldsymbol{B})\boldsymbol{Y}^{-1}$. 若 $\boldsymbol{B}$ 比较简单, 则 $f(\boldsymbol{B})$ 常常可以直接计算. 在 11.1 节中, 利用 Jordan 分解与 Schur 分解阐明了这一点. 利用后一种分解, 计算 $f(\boldsymbol{A})$ 的算法更稳定, 这是不足为奇的.

处理矩阵函数问题的另一类方法是用一个容易计算的函数 $g(\boldsymbol{A})$ 来逼近所求的 $f(\boldsymbol{A})$. 例如, $g$ 可以是 $f$ 的 Taylor 展开式的前若干项. 11.2 节给出了这类逼近问题的误差界.

在 11.3 节, 我们讨论了一个特殊的但非常重要的问题: 计算矩阵指数 $\mathrm{e}^{\boldsymbol{A}}$.

**预备知识**

阅读本章需要掌握第 1, 2, 3, 7, 8 章的知识. 本章各节间的关系如下:

$$11.1\text{节}\rightarrow 11.2\text{节}\rightarrow 11.3\text{节}$$

补充的参考文献有: Mirsky(1955), Gantmacher(1959), Bellman(1969) 及 Horn and Johnson(1991). 对本章非常重要的 MATLAB函数有: `expm`, `expm1`, `expm2`, `empm3`, `logm`, `sqrtm`和`funm`.

## 11.1　特征值方法

给定一 $n\times n$ 矩阵 $\boldsymbol{A}$ 和一标量函数 $f(z)$, 有好几种方法定义*矩阵函数*$f(\boldsymbol{A})$. 一种很不正式的定义是在 $f(z)$ 的表达式中用 $\boldsymbol{A}$ 代替 $z$. 例如, 若 $p(z)=1+z$ 及 $r(z)=\left(1-\dfrac{z}{2}\right)^{-1}\left(1+\dfrac{z}{2}\right), z\neq 2$, 则自然地定义 $p(\boldsymbol{A})$ 和 $r(\boldsymbol{A})$ 如下:

$$p(\boldsymbol{A})=\boldsymbol{I}+\boldsymbol{A},$$

$$r(\boldsymbol{A})=\left(\boldsymbol{I}-\frac{\boldsymbol{A}}{2}\right)^{-1}\left(\boldsymbol{I}+\frac{\boldsymbol{A}}{2}\right),\quad 2\notin\lambda(\boldsymbol{A}).$$

用 $\boldsymbol{A}$ 代替 $z$, 也可用于超越函数, 例如:

$$\mathrm{e}^{\boldsymbol{A}}=\sum_{k=0}^{\infty}\frac{\boldsymbol{A}^k}{k!}.$$

然而, 为了使以后的算法更精确, 我们需要 $f(\boldsymbol{A})$ 的更精确的定义.

### 11.1.1 定义

有许多种方法来严格地建立矩阵函数的概念, 见 Rinehart(1955). 最完美的方法也许是利用周线积分. 假设 $f(z)$ 在一个闭周线 $\Gamma$ 的内部解析, 且 $\Gamma$ 包围了 $\lambda(\boldsymbol{A})$. 我们定义 $f(\boldsymbol{A})$ 是矩阵:

$$f(\boldsymbol{A}) = \frac{1}{2\pi \mathrm{i}} \oint_{\Gamma} f(z)(z\boldsymbol{I} - \boldsymbol{A})^{-1} \mathrm{d}z. \tag{11.1.1}$$

显然, 这一定义是 Cauchy 积分定理的矩阵形式. 上面的积分也可逐个元素定义:

$$f(\boldsymbol{A}) = (f_{kj}) \Rightarrow f_{kj} = \frac{1}{2\pi \mathrm{i}} \oint_{\Gamma} f(z) e_k^T (z\boldsymbol{I} - \boldsymbol{A})^{-1} e_j \mathrm{d}z.$$

注意, 矩阵 $(z\boldsymbol{I} - \boldsymbol{A}^{-1})$ 的元素在 $\Gamma$ 上解析, 并且只要 $f(z)$ 在 $\lambda(\boldsymbol{A})$ 的一个邻域内解析, $f(\boldsymbol{A})$ 就有定义.

### 11.1.2 Jordan 特征

定义 (11.1.1) 虽然在计算上用处不大, 但它可用来导出 $f(\boldsymbol{A})$ 更多的有用的特征. 例如, 若 $f(\boldsymbol{A})$ 有定义, 且

$$\boldsymbol{A} = \boldsymbol{X}\boldsymbol{B}\boldsymbol{X}^{-1} = \boldsymbol{X}\mathrm{diag}(\boldsymbol{B}_1, \cdots, \boldsymbol{B}_p)\boldsymbol{X}^{-1}, \quad \boldsymbol{B}_i \in \mathbb{C}^{n_i \times n_i}$$

则易证

$$f(\boldsymbol{A}) = \boldsymbol{X} f(\boldsymbol{B}) \boldsymbol{X}^{-1} = \boldsymbol{X}\mathrm{diag}(f(\boldsymbol{B}_1), \cdots, f(\boldsymbol{B}_p))\boldsymbol{X}^{-1}. \tag{11.1.2}$$

当 $\boldsymbol{B}$ 为 Jordan 块时, 我们可得如下结果.

**定理 11.1.1** 设 $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} = \mathrm{diag}(\boldsymbol{J}_1, \boldsymbol{J}_2, \cdots, \boldsymbol{J}_p)$ 为 $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ 的 Jordan 标准型 (JCF), 其中

$$\boldsymbol{J}_i = \begin{bmatrix} \lambda_i & 1 & \cdots & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & \lambda_i \end{bmatrix}$$

为 $m_i \times m_i$ Jordan 块. 如果 $f(z)$ 在包含 $\lambda(\boldsymbol{A})$ 的一个开集上解析, 则 $f(\boldsymbol{A}) = \boldsymbol{X}\mathrm{diag}(f(\boldsymbol{J}_1), \cdots, f(\boldsymbol{J}_p))\boldsymbol{X}^{-1}$, 其中

$$f(\boldsymbol{J}_i) = \begin{bmatrix} f(\lambda_i) & f^{(1)}(\lambda_i) & \cdots & \cdots & \dfrac{f^{(m_i-1)}(\lambda_i)}{(m_i-1)!} \\ 0 & f(\lambda_i) & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & f^{(1)}(\lambda_i) \\ 0 & \cdots & \cdots & \cdots & f(\lambda_i) \end{bmatrix}.$$

**证明** 根据定理之前的叙述, 只要观察 $f(G)$ 的形式就够了, 其中 $G=\lambda I+E$ $(E=(\delta_{i,j-1}))$ 为 $q\times q$ Jordan 块. 假设 $(zI-G)$ 非奇异. 因为

$$(zI-G)^{-1}=\sum_{k=0}^{q-1}\frac{E^k}{(z-\lambda)^{k+1}},$$

由 Cauchy 积分定理可知

$$f(G)=\sum_{k=0}^{q-1}\left[\frac{1}{2\pi i}\oint_{\Gamma}\frac{f(z)}{(z-\lambda)^{k+1}}dz\right]E^k=\sum_{k=0}^{q-1}\frac{f^{(k)}(\lambda)}{k!}E^k.$$

注意 $E^k=(\delta_{i,j-k})$ 即知定理成立. □

**推论 11.1.2** *如果 $A\in\mathbb{C}^{n\times n}$, $A=X\mathrm{diag}(\lambda_1,\ldots,\lambda_n)X^{-1}$, $f(A)$ 有定义, 则*

$$f(A)=X\mathrm{diag}(f(\lambda_1),\cdots,f(\lambda_n))X^{-1}.$$

**证明** 所有的 Jordan 块都是 $1\times 1$ 的. □

这些结果表明了 $f(A)$ 与 $A$ 的特征系统之间的紧密联系. 不幸的是, 除非 $A$ 能够用一个良态的特征向量矩阵对角化, 否则用 Jordan 标准型处理矩阵函数在计算上是不可靠的. 事实上, 因为必须解涉及矩阵 $X$ 的线性方程组, 数量级为 $u\kappa_2(X)$ 的舍入误差破坏计算结果是预料之中的. 下面的例子说明计算矩阵函数时应避免病态的相似变换.

**例 11.1.1** 如果

$$A=\begin{bmatrix}1+10^{-5} & 1\\ 0 & 1-10^{-5}\end{bmatrix},$$

则任一特征向量的矩阵都是

$$X=\begin{bmatrix}1 & -1\\ 0 & 2(1-10^{-5})\end{bmatrix}$$

的列加权, 故其 2 范数条件数量级为 $10^5$. 利用机器精度为 $u\approx 10^{-7}$ 的计算机运算, 我们有

$$fl[X^{-1}\mathrm{diag}(e^{(1+10^{-5})},e^{(1-10^{-5})})X]=\begin{bmatrix}2.718\,307 & 2.750\,000\\ 0.000\,000 & 2.718\,254\end{bmatrix},$$

而

$$e^{A}=\begin{bmatrix}2.718\,309 & 2.718\,282\\ 0.000\,000 & 2.718\,255\end{bmatrix}.$$

### 11.1.3 利用 Schur 分解的方法

利用 Schur 分解处理矩阵函数可避免用 Jordan 方法所带来的一些困难. 如果 $A=QTQ^{\mathrm{H}}$ 是 $A$ 的 Schur 分解, 则

$$f(A)=Qf(T)Q^{\mathrm{H}}.$$

为了使其有效, 我们需要一个计算上三角形矩阵的函数的算法. 不幸的是, 正如以下定理所示, $f(T)$ 的显式表达是非常复杂的.

**定理 11.1.3** 设 $T=(t_{ij})$ 是 $n\times n$ 上三角形矩阵, 特征值 $\lambda_i=t_{ii}$, 且 $f(T)$ 有定义. 如果 $f(T)=(f_{ij})$, 则当 $i>j$ 时 $f_{ij}=0$, 当 $i=j$ 时 $f_{ij}=f(\lambda_i)$, 而当 $i<j$ 时,

$$f_{ij}=\sum_{(s_0,\cdots,s_k)\in S_{ij}} t_{s_0,s_1}t_{s_1,s_2}\cdots t_{s_{k-1},s_k}f[\lambda_{s_0},\cdots,\lambda_{s_k}],$$

其中 $S_{ij}$ 为所有以 $i$ 开头且以 $j$ 结尾的严格单调递增整数序列组成的集合, $f[\lambda_{s_0},\cdots,\lambda_{s_k}]$ 是 $f$ 在 $\{\lambda_{s_0},\cdots,\lambda_{s_k}\}$ 的 $k$ 阶均差.

**证明** 见 Descloux(1963), Davis(1973) 或 Van Loan(1975). □

用定理 11.1.3 计算 $f(T)$ 需 $O(2^n)$ 个 flop. 幸运的是, Parlett(1974) 导出了一个计算矩阵 $F=f(T)$ 的严格上三角部分的完美递推公式, 它仅需 $2n^3/3$ 个 flop, 而且可由可交换结果

$$FT=TF \tag{11.1.3}$$

导出. 事实上, 比较此方程两边的 $(i,j)$ 元可知:

$$\sum_{k=i}^{j} f_{ik}t_{kj}=\sum_{k=i}^{j} t_{ik}f_{kj},\quad j>i.$$

因此, 若 $t_{ij}\neq t_{ij}$, 则

$$f_{ij}=t_{ij}\frac{f_{jj}-f_{ii}}{t_{jj}-t_{ii}}+\sum_{k=i+1}^{j-1}\frac{t_{ik}f_{kj}-f_{ik}t_{kj}}{t_{jj}-t_{ii}}. \tag{11.1.4}$$

从此可知, $f_{ij}$ 是矩阵 $F$ 中它的左边和下边的邻近元素的线性组合. 例如, 元素 $f_{25}$ 依赖于 $f_{22},f_{23},f_{24},f_{55},f_{45},f_{35}$ 的值. 因此, $F$ 的整个上三角部分都可以算出, 从对角元 $f(t_{11}),\cdots,f(t_{nn})$ 开始, 每次能计算出一条超对角线. 整个算法如下.

**算法 11.1.1** 本算法计算出矩阵函数 $F=f(T)$, 其中 $T$ 为上三角形矩阵且特征值互不相同, $f$ 在 $\lambda(T)$ 上有定义.

```
for i = 1 : n
    f_ii = f(t_ii)
end
for p = 1 : n-1
    for i = 1 : n-p
        j = i + p
        s = t_ij(f_jj - f_ii)
        for k = i+1 : j-1
            s = s + t_ik f_kj - f_ik t_kj
        end
        f_ij = s/(t_jj - t_ii)
    end
end
```

本算法需要 $2n^3/3$ 个 flop. 假设 $\boldsymbol{T} = \boldsymbol{Q}\boldsymbol{A}\boldsymbol{Q}^{\mathrm{H}}$ 是 $\boldsymbol{A}$ 的 Schur 分解, 则 $f(\boldsymbol{A}) = \boldsymbol{Q}\boldsymbol{F}\boldsymbol{Q}^{\mathrm{H}}$, 其中 $\boldsymbol{F} = f(\boldsymbol{T})$. 显然, 计算 $f(\boldsymbol{A})$ 的大多数工作量都花在 Schur 分解上, 除非函数 $f$ 的计算代价极其昂贵.

**例 11.1.2** 如果

$$\boldsymbol{T} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 3 & 4 \\ 0 & 0 & 5 \end{bmatrix},$$

$f(z) = (1+z)/z$, 则 $\boldsymbol{F} = (f_{ij}) = f(\boldsymbol{T})$ 的定义如下:

$$\begin{aligned}
f_{11} &= (1+1)/1 = 2, \\
f_{22} &= (1+3)/3 = 4/3, \\
f_{33} &= (1+5)/5 = 6/5, \\
f_{12} &= t_{12}(f_{22} - f_{11})/(t_{22} - t_{11}) = -2/3, \\
f_{23} &= t_{23}(f_{33} - f_{22})/(t_{33} - t_{22}) = -4/15, \\
f_{13} &= [t_{13}(f_{33} - f_{11}) + (t_{12}f_{23} - t_{12}t_{23})]/(t_{33} - t_{11}) = -1/15.
\end{aligned}$$

### 11.1.4 分块 Schur 方法

如果矩阵 $\boldsymbol{A}$ 有相近的或相同的特征值, 则算法 11.1.1 效果很差. 在此情形下, 可推荐用算法 11.1.1 的分块形式. 我们大致介绍一下归功于 Parlett(1974a) 的这样一个算法. 第一步是在 Schur 分解中选择一个恰当的 $\boldsymbol{Q}$, 使得 $\boldsymbol{A}$ 的相近或相同的特征值集中在 $\boldsymbol{T}$ 的对角块 $\boldsymbol{T}_{11}, \cdots, \boldsymbol{T}_{pp}$ 中. 确切地说, 我们必须要计算一种划分:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{T}_{12} & \cdots & \boldsymbol{T}_{1p} \\ \boldsymbol{0} & \boldsymbol{T}_{22} & \cdots & \boldsymbol{T}_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{T}_{pp} \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}_{11} & \boldsymbol{F}_{12} & \cdots & \boldsymbol{F}_{1p} \\ \boldsymbol{0} & \boldsymbol{F}_{22} & \cdots & \boldsymbol{F}_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{F}_{pp} \end{bmatrix},$$

其中 $\lambda(\boldsymbol{T}_{ii}) \cap \lambda(\boldsymbol{T}_{jj}) \neq \varnothing, i \neq j$. 用 7.6 节中的方法可以决定块的大小.

下一步是计算子矩阵 $\boldsymbol{F}_{ii} = f(\boldsymbol{T}_{ii}), i = 1:p$. 既然 $\boldsymbol{T}_{ii}$ 的特征值很相近, 那么这些 $f(\boldsymbol{T}_{ii})$ 的计算需要特殊的方法. (在以下两节中, 我们将讨论几种方法.) 一旦 $\boldsymbol{F}$ 的对角块已知了, $\boldsymbol{F}$ 的严格上三角块也可递推算出, 就像所有的块都是标量一样. 比较 $\boldsymbol{F}\boldsymbol{T} = \boldsymbol{T}\boldsymbol{F}$ 中 $i < j$ 的块 $(i, j)$, 可得 (11.1.4) 的如下推广, 它是确定递推关系的方程:

$$\boldsymbol{F}_{ij}\boldsymbol{T}_{jj} - \boldsymbol{T}_{ii}\boldsymbol{F}_{ij} = \boldsymbol{T}_{ij}\boldsymbol{F}_{jj} - \boldsymbol{F}_{ii}\boldsymbol{T}_{ij} + \sum_{k=i+1}^{j-1} (\boldsymbol{T}_{ik}\boldsymbol{F}_{kj} - \boldsymbol{F}_{ik}\boldsymbol{T}_{kj}). \tag{11.1.5}$$

如果每次要算出 $\boldsymbol{F}$ 的一个超对角块 $\boldsymbol{F}_{ij}$, 则这是一个右边已知, 未知量为块 $F_{ij}$ 之元素的线性方程组. 它可以用 Bartels-Stewart 算法 (算法 7.6.2) 来求解.

计算实矩阵的实函数时, 这里介绍的计算 $f(\boldsymbol{A})$ 的分块 Schur 方法是很有用的. 在计算了实 Schur 分解 $\boldsymbol{A}=\boldsymbol{QTQ}^{\mathrm{T}}$ 后, 可用上面的分块算法沿着 $\boldsymbol{T}$ 的对角线来处理这些 $2\times 2$ 块的计算问题.

## 习 题

**11.1.1** 利用定义 (11.1.1) 证明: (a) $\boldsymbol{A}f(\boldsymbol{A})=f(\boldsymbol{A})\boldsymbol{A}$, (b) 当 $\boldsymbol{A}$ 是上三角形矩阵时, $f(\boldsymbol{A})$ 也是上三角形矩阵, (c) 当 $\boldsymbol{A}$ 是 Hermite 矩阵时, $f(\boldsymbol{A})$ 也是 Hermite 矩阵.

**11.1.2** 改写算法 11.1.1, 使得 $f(\boldsymbol{T})$ 一列一列来计算.

**11.1.3** 设 $\boldsymbol{A}=\boldsymbol{X}\mathrm{diag}(\lambda_i)\boldsymbol{X}^{-1}$, 其中 $\boldsymbol{X}=[\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n]$ 和 $\boldsymbol{X}^{-1}=[\boldsymbol{y}_1,\cdots,\boldsymbol{y}_n]^{\mathrm{H}}$. 证明: 如果 $f(\boldsymbol{A})$ 有定义, 则

$$f(\boldsymbol{A})=\sum_{k=1}^{n}f(\lambda_i)\boldsymbol{x}_i\boldsymbol{y}_i^{\mathrm{H}}.$$

**11.1.4** 证明:

$$\boldsymbol{T}=\begin{bmatrix}\boldsymbol{T}_{11} & \boldsymbol{T}_{12}\\ \boldsymbol{0} & \boldsymbol{T}_{22}\end{bmatrix}\begin{matrix}p\\ q\end{matrix}\Rightarrow f(\boldsymbol{T})=\begin{bmatrix}\boldsymbol{F}_{11} & \boldsymbol{F}_{12}\\ \boldsymbol{0} & \boldsymbol{F}_{22}\end{bmatrix}\begin{matrix}p\\ q\end{matrix}$$
$$\qquad\quad p\quad\ q \qquad\qquad\qquad\quad p\quad\ q$$

其中 $\boldsymbol{F}_{11}=f(\boldsymbol{T}_{11}),\boldsymbol{F}_{22}=f(\boldsymbol{T}_{22})$, 这里假定 $f(\boldsymbol{T})$ 有定义.

## 本节注释与参考文献

本书中介绍的 $f(\boldsymbol{A})$ 的周线积分表示由于其一般性在泛函分析中很有用, 见:

N. Dunford and J. Schwartz (1958). *Linear Operators, Part I*, Interscience, New York.

正如我们所说, 也可用 $f(\boldsymbol{A})$ 的其他定义. 但是, 对于在实际中通常出现的矩阵函数, 所有这些定义都是等价的. 见:

R. F. Rinehart(1955). "The Equivalence of Definitions of a Matric Function," *Amer Math. Monthy* 62, 395–414.

关于 Jordan 表示的各个方面详见:

J. S. Frame(1964). "Matrix Functions and Applications, Part II," *IEEE Spectrum* 1 *(April)*, 102–131.

J. S. Frame(1964). "Matrix Functions and Applications, Part Ⅳ," *IEEE Spectrum* 1 *(June)*, 123-131.

下面的文献涉及 Schur 分解及其与 $f(\boldsymbol{A})$ 问题的联系:

D. Davis(1973). "Explioit Functional Calculus" *Lin. Alg. and Its Applic.*6,193–199.

J. Descloux (1963). "Bounds for the Spectral Morm of Functions of Matrices, "*Numer. Math.*5,185–190.

C. F. Van Loan(1975). "A Study of the Matrix Exponential, " Numerical Analysis Report No. 10, Dept. of Maths., University of Manchester, England.

算法 11.1 以及当其用于有相近的或相同的特征值的矩阵时的各种各样的计算困难可见下列文献的讨论:

B. N. Parlett(1976). "A Recurrence Among the Elements of Functions of Triangular Matrices," *Lin. Alg. and Its Applic.* 14, 117–121.

如果 $\boldsymbol{A}$ 可约化成块对角形式 (7.6.3 节), 则用于 $f(\boldsymbol{A})$ 问题的 Jordan 方法与 Schur 方法可达成妥协, 见:

B. Kågström(1977). "Numerical Computation of Matrix Functions," Department of Information Processing Report UMINF-58.77, University of Ümea, Sweden.

关于矩阵函数对扰动的敏感性的讨论可见:

C. S. Kenney and A. J. Laub(1989). "Condition Estimates for Matrix Functions," *SIAMJ. Matrix Anal. Appl.* 10, 191–209.

C. S Kenney and A. J. Laub(1994). "Small-Sample Statistical Condition Estimates for General Matrix Functions," *SIAM J. Sci. Comp.* 15, 36–61.

本章的一个主题是, 若 $\boldsymbol{A}$ 非正规, 则计算 $f(\boldsymbol{A})$ 远不仅是在 $\lambda(\boldsymbol{A})$ 上计算 $f(z)$, 伪特征值是理解该现象的一种方法, 见:

L. N. Trefethen(1992). "Pseudospectra of Matrices," in *Numerical Analysis* 1991, D. F. Griffiths and G. A. Watson(eds), Longman Scientific & Technical, Harlow, Essex, UK.

在 11.3.4 节将有更详细的讨论.

## 11.2 逼 近 法

本节, 我们讨论一类初看起来不涉及特征值的计算矩阵函数的方法. 这类方法的基本思想是, 如果 $g(z)$ 在 $\lambda(\boldsymbol{A})$ 上逼近 $f(z)$, 则 $g(\boldsymbol{A})$ 逼近 $f(\boldsymbol{A})$, 例如:

$$\mathrm{e}^{\boldsymbol{A}} \approx \boldsymbol{I} + \boldsymbol{A} + \frac{\boldsymbol{A}^2}{2!} + \frac{\boldsymbol{A}^3}{3!} + \cdots + \frac{\boldsymbol{A}^q}{q!}.$$

首先我们用矩阵函数的 Jordan 表示和 Schur 表示来估计 $\|f(\boldsymbol{A}) - g(\boldsymbol{A})\|$ 的界. 之后再讨论一下矩阵多项式的计算.

### 11.2.1 Jordan 分析

矩阵函数的 Jordan 表示 (定理 11.1.1) 可用来给出 $g(\boldsymbol{A})$ 逼近 $f(\boldsymbol{A})$ 的误差界.

**定理 11.2.1** 设 $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ 的 Jordan 标准型为

$$\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X} = \mathrm{diag}(\boldsymbol{J}_1, \boldsymbol{J}_2, \cdots, \boldsymbol{J}_p),$$

其中

$$\boldsymbol{J}_i = \begin{bmatrix} \lambda_i & 1 & \cdots & \cdots & 0 \\ 0 & \lambda_i & 1 & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & \lambda_i \end{bmatrix}$$

为 $m_i \times m_i$ Jardan 块. 如果 $f(z)$ 和 $g(z)$ 在包含 $\lambda(\boldsymbol{A})$ 的一开集上解析, 则

$$\|f(\boldsymbol{A}) - g(\boldsymbol{A})\|_2 \leqslant \kappa_2(\boldsymbol{X}) \max_{\substack{1\leqslant i\leqslant p \\ 0\leqslant r\leqslant m_i-1}} m_i \frac{|f^{(r)}(\lambda_i) - g^{(r)}(\lambda_i)|}{r!}.$$

**证明** 令 $h(z)=f(z)-g(z)$, 则

$$\|f(\boldsymbol{A})-g(\boldsymbol{A})\|_2=\|\boldsymbol{X}\mathrm{diag}(h(\boldsymbol{J}_1),\cdots,h(\boldsymbol{J}_p))\boldsymbol{X}^{-1}\|_2\leqslant\kappa_2(x)\max_{1\leqslant i\leqslant p}\|h(\boldsymbol{J}_i)\|_2.$$

应用定理 11.1.1 和等式 (2.3.8), 可得

$$\|h(\boldsymbol{J}_i)\|_2\leqslant m_i\cdot\max_{0\leqslant r\leqslant m_i-1}\frac{|h^{(r)}(\lambda_i)|}{r!},$$

由此可知定理成立. □

### 11.2.2 Schur 分析

如果不用 Jordan 分解而用 Schur 分解, 则可得另一个误差界.

**定理 11.2.2** 设 $\boldsymbol{A}\in\mathbb{C}^{n\times n}$ 的 Schur 分解为 $\boldsymbol{Q}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{T}=\mathrm{diag}(\lambda_i)+\boldsymbol{N}$, 其中 $\boldsymbol{N}$ 为 $\boldsymbol{T}$ 的严格上三角部分. 如果 $f(z)$ 和 $g(z)$ 在一个内部包含 $\lambda(\boldsymbol{A})$ 的闭凸集 $\Omega$ 上解析, 则

$$\|f(\boldsymbol{A})-g(\boldsymbol{A})\|_F\leqslant\sum_{r=0}^{n-1}\delta_r\frac{\|\,|\boldsymbol{N}|^r\,\|_F}{r!},$$

其中 $\delta_r=\sup\limits_{z\in\Omega}|f^{(r)}(z)-g^{(r)}(z)|$.

**证明** 令 $h(z)=f(z)-g(z),\boldsymbol{H}=(h_{ij})=h(\boldsymbol{A})$. 记 $S_{ij}^{(r)}$ 为所有满足 $s_0=i,s_r=j$ 的严格递增整数序列 $(s_0,s_1,\cdots,s_r)$ 组成的集合. 注意到 $S_{ij}=\bigcup\limits_{r=1}^{j-i}S_{ij}^{(r)}$, 从定理 11.1.3 可知, 对所有的 $i<j$, 有

$$h_{ij}=\sum_{r=1}^{j-1}\sum_{s\in S_{ij}^{(r)}}n_{s_0,s_1}n_{s_1,s_2}n_{s_2,s_3}\cdots n_{s_{r-1},s_r}h[\lambda_{s_0},\cdots,\lambda_{s_r}].$$

即然 $\Omega$ 是凸的且 $h$ 解析, 那么

$$|h[\lambda_{s_0},\cdots,\lambda_{s_r}]|\leqslant\sup_{z\in\Omega}\frac{|h^{(r)}(z)|}{r!}=\frac{\delta_r}{r!}.\tag{11.2.1}$$

而且, 若记 $|\boldsymbol{N}|^r=(n_{ij}^{(r)}),r\geqslant1$, 则可证:

$$n_{ij}^{(r)}=\begin{cases}0 & j<i+r,\\ \displaystyle\sum_{s\in S_{ij}^{(r)}}|n_{s_0,s_1}n_{s_1,s_2}\cdots n_{s_{r-1},s_r}|, & j\geqslant i+r\end{cases}\tag{11.2.2}$$

在 $h_{ij}$ 的表达式两边取绝对值, 然后利用 (11.2.1) 和 (11.2.2), 可知定理成立. □

上面定理中的界表明, 仅仅在 $\boldsymbol{A}$ 的谱 $\lambda(\boldsymbol{A})$ 上逼近 $f(\boldsymbol{A})$ 是不够的. 特别地, 如果 $\boldsymbol{A}$ 的特征矩阵是病态的或者它同正规矩阵的分离度很大, 则 $f(\boldsymbol{A})$ 与 $g(\boldsymbol{A})$ 的差可能会比 $|f(z)-g(z)|$ 在 $\lambda(\boldsymbol{A})$ 上的极大值大得多. 因此, 虽然逼近方法可以避免特征值的计算, 但它受到 $\boldsymbol{A}$ 的特征矩阵的结构的影响. 这一点我们下一节将继续讨论.

**例 11.2.1** 假设

$$A = \begin{bmatrix} -0.01 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0.01 \end{bmatrix}.$$

若 $f(z) = e^z, g(z) = 1 + z + z^2/2$, 则无论是 Frobenius 范数还是 2 范数, 都有 $\|f(A) - g(A)\| \approx 10^{-5}$. 因为 $\kappa_2(X) \approx 10^{-7}$, 用定理 11.2.1 估计的误差为 $O(1)$, 比较保守. 另一方面, 用 Schur 分解估计的误差为 $O(10^{-2})$.

### 11.2.3 Taylor 逼近

一种逼近矩阵函数 (例如 $e^A$) 的常用方法是用截断的 Taylor 级数. 一个矩阵函数 $f(A)$ 存在 Taylor 级数展开的条件是很容易找到的.

**定理 11.2.3** 如果 $f(z)$ 在包含 $\lambda(A)$ 的一个圆盘上有 Taylor 级数展开:

$$f(z) = \sum_{k=0}^{\infty} c_k z^k,$$

则

$$f(A) = \sum_{k=0}^{\infty} c_k A^k.$$

**证明** 我们对 $A$ 是可对角化的情形来证明定理. 在习题 11.2.1 中, 给出提示说明无此假设如何处理. 设 $X^{-1}AX = D = \text{diag}(\lambda_1, \cdots, \lambda_n)$, 应用推论 11.1.2 有

$$\begin{aligned} f(A) &= X\text{diag}(f(\lambda_1), \cdots, f(\lambda_n))X^{-1} \\ &= X\text{diag}\Big(\sum_{k=0}^{\infty} c_k\lambda_1^k, \cdots, \sum_{k=0}^{\infty} c_k\lambda_n^k\Big)X^{-1} \\ &= X\Big(\sum_{k=0}^{\infty} c_k D^k\Big)X^{-1} \\ &= \sum_{k=0}^{\infty} c_k(XDX^{-1})^k = \sum_{k=0}^{\infty} c_k A^k. \end{aligned}$$

□

几个重要的超越矩阵函数有特别简单的级数展开:

$$\begin{aligned} \ln(I - A) &= \sum_{k=1}^{\infty} \frac{A^k}{k}, \quad |\lambda| < 1, \lambda \in \lambda(A), \\ \sin(A) &= \sum_{k=0}^{\infty} (-1)^k \frac{A^{2k+1}}{(2k+1)!}, \\ \cos(A) &= \sum_{k=0}^{\infty} (-1)^k \frac{A^{2k}}{(2k)!}. \end{aligned}$$

下面的定理给出了用截断的 Taylor 级数来逼近矩阵函数的误差界.

**定理 11.2.4** 给定 $A \in \mathbb{C}^{n\times n}$ 如果 $f(z)$ 在包含 $\lambda(A)$ 的开圆盘上有 Taylor 展开:

$$f(z) = \sum_{k=0}^{\infty} \alpha_k z^k,$$

则有以下估计

$$\left\|f(\boldsymbol{A}) - \sum_{k=0}^{q} \alpha_k \boldsymbol{A}^k\right\|_2 \leqslant \frac{n}{(q+1)!} \max_{0\leqslant s\leqslant 1} \|\boldsymbol{A}^{q+1} f^{(q+1)}(\boldsymbol{A}s)\|_2.$$

**证明** 定义矩阵 $E(s)$ 如下:

$$f(\boldsymbol{A}s) = \sum_{k=0}^{q} \alpha_k (\boldsymbol{A}s)^k + E(s), \quad 0 \leqslant s \leqslant 1. \tag{11.2.3}$$

若以 $f_{ij}(s)$ 来表示 $f(\boldsymbol{A}s)$ 的 $(i,j)$ 元，则它肯定是解析的，所以有

$$f_{ij}(s) = \left(\sum_{k=0}^{q} \frac{f_{ij}^{(k)}(0)}{k!} s^k\right) + \frac{f_{ij}^{(q+1)}(\varepsilon_{ij})}{(q+1)!} s^{q+1}, \tag{11.2.4}$$

其中 $0 \leqslant \varepsilon_{ij} \leqslant s < 1$.

在 (11.2.3) 和 (11.2.4) 中比较 $s$ 的幂, 可知矩阵 $E(s)$ 的 $(i,j)$ 元 $e_{ij}(s)$ 有如下形式:

$$e_{ij}(s) = \frac{f_{ij}^{(q+1)}(\varepsilon_{ij})}{(q+1)!} s^{q+1}.$$

现知 $f_{ij}^{(q-1)}(s)$ 为矩阵 $\boldsymbol{A}^{q+1} f^{(q+1)}(\boldsymbol{A}s)$ 的 $(i,j)$ 元，因此

$$|e_{ij}(s)| \leqslant \max_{0\leqslant s\leqslant 1} \frac{f_{ij}^{(q+1)}(s)}{(q+1)!} \leqslant \max_{0\leqslant s\leqslant 1} \frac{\|\boldsymbol{A}^{q+1} f^{(q+1)}(\boldsymbol{A}s)\|_2}{(q+1)!}.$$

应用 (2.3.8) 式, 即得定理结论. □

**例 11.2.2** 若 $\boldsymbol{A} = \begin{bmatrix} -49 & 24 \\ -64 & 31 \end{bmatrix}$, 则

$$\mathrm{e}^{\boldsymbol{A}} = \begin{bmatrix} -0.735\,759 & 0.055\,181\,9 \\ -1.471\,518 & 1.103\,638 \end{bmatrix}.$$

当 $q = 59$ 时, 定理 11.2.4 表明误差界为

$$\|\mathrm{e}^{\boldsymbol{A}} - \sum_{k=0}^{q} \frac{\boldsymbol{A}^k}{k!}\|_2 \leqslant \frac{n}{(q+1)!} \max_{0\leqslant s\leqslant 1} \|\boldsymbol{A}^{q+1} \mathrm{e}^{\boldsymbol{A}s}\|_2 \leqslant 10^{-60}.$$

然而, 当 $\mathbf{u} \approx 10^{-7}$ 时, 发现

$$fl\left(\sum_{k=0}^{59} \frac{\boldsymbol{A}^k}{k!}\right) = \begin{bmatrix} -22.258\,80 & -1.432\,276\,6 \\ -61.499\,31 & -3.474\,280 \end{bmatrix}$$

问题在于部分和中的有一些很大的元素. 例如, 在 $\boldsymbol{I} + \boldsymbol{A} + \cdots + \boldsymbol{A}^{17}/17!$ 中, 有的元素竟达到 $O(10^7)$, 然而机器精度只有 $10^{-7}$, 所以舍入误差比解的范数要大得多.

例 11.2.2 说明了用截断的 Taylor 级数来逼近矩阵函数的一个缺点是只有在原点附近它才有意义. 有时候, 通过变换可克服这一点. 例如, 重复应用倍角公式:

$$\cos(2A) = 2\cos^2(\boldsymbol{A}) - \boldsymbol{I}, \quad \sin(2\boldsymbol{A}) = 2\sin(\boldsymbol{A})\cos(\boldsymbol{A}),$$

可用适当的截断 Taylor 级数逼近"组装"矩阵的正弦值与余弦值:

$S_0 = \sin(\boldsymbol{A}/2^k)$的 Taylor 逼近

$C_0 = \cos(\boldsymbol{A}/2^k)$的 Taylor 逼近

**for** $j = 1 : k$

$\quad S_j = 2S_{j-1}C_{j-1}$

$\quad C_j = 2C_{j-1}^2 - I$

**end**

这里 $k$ 是一个适当选取的正整数, 比如 $||\boldsymbol{A}||_\infty = 2^k$. 参阅 Serkin and Blalock(1979).

### 11.2.4 计算矩阵多项式

因为在超越矩阵函数的逼近中经常涉及计算多项式, 所以有必要详细讨论多项式

$$p(\boldsymbol{A}) = b_0\boldsymbol{I} + b_1\boldsymbol{A} + \cdots + b_q\boldsymbol{A}^q$$

的计算问题, 其中 $b_0, \cdots, b_q$ 为已知的实数. 最明显的办法是用 Horner 技巧.

**算法 11.2.1** 给定一矩阵 $A$ 和向量 $b(0:q)$, 则本算法计算多项式:

$\boldsymbol{F} = b_q\boldsymbol{A}^q + \cdots + b_1\boldsymbol{A} + b_0\boldsymbol{I}$

$F = b_qA + b_{q-1}I$

**for** $k = q-2 : -1 : 0$

$\quad F = AF + b_kI$

**end**

该算法需要 $q-1$ 次矩阵乘法. 然而, 和标量的情形不一样, 这种求和过程并不是最好的. 为什么呢? 以 $q = 9$ 为例来说明. 注意到:

$$\begin{aligned} p(\boldsymbol{A}) =& \boldsymbol{A}^3(\boldsymbol{A}^3(b_9\boldsymbol{A}^3 + (b_8\boldsymbol{A}^2 + b_7\boldsymbol{A} + b_6\boldsymbol{I})) \\ &+ (b_5\boldsymbol{A}^2 + b_4\boldsymbol{A} + b_3\boldsymbol{I}) + b_2\boldsymbol{A}^2 + b_1\boldsymbol{A} + b_0\boldsymbol{I}. \end{aligned}$$

因此, 只用四次矩阵乘法就可算出 $F = p(\boldsymbol{A})$:

$$\boldsymbol{A}_2 = \boldsymbol{A}^2$$

$$\boldsymbol{A}_3 = \boldsymbol{A}\boldsymbol{A}_2$$

$$F_1 = b_9\boldsymbol{A}_3 + b_8\boldsymbol{A}_2 + b_7\boldsymbol{A} + b_6\boldsymbol{I}$$

$$F_2 = \boldsymbol{A}_3F_1 + b_5\boldsymbol{A}_2 + b_4\boldsymbol{A} + b_3\boldsymbol{I}$$

$$F = \boldsymbol{A}_3F_2 + b_2\boldsymbol{A}_2 + b_1\boldsymbol{A} + b_0\boldsymbol{I}$$

一般来说，若 $s$ 是满足 $1 \leqslant s \leqslant \sqrt{q}$ 的任一整数，则

$$p(\boldsymbol{A}) = \sum_{k=0}^{r} \boldsymbol{B}_k (\boldsymbol{A}^s)^k, \quad r = \text{floor}(q/s), \tag{11.2.5}$$

其中

$$\boldsymbol{B}_k = \begin{cases} b_{sk+s-1}\boldsymbol{A}^{s-1} + \cdots + b_{sk+1}\boldsymbol{A} + b_{sk}\boldsymbol{I}, & k = 0 : r-1, \\ b_q\boldsymbol{A}^{q-sr} + \cdots + b_{sr+1}\boldsymbol{A} + b_k\boldsymbol{I}, & k = r. \end{cases}$$

一旦 $\boldsymbol{A}^2, \cdots, \boldsymbol{A}^s$ 算出, Horher 规则可用于 (11.2.5) 式. 所以, $p(\boldsymbol{A})$ 可以只用 $s+r-1$ 次矩阵乘法就得到. 当 $s = \text{floor}(\sqrt{q})$ 时, 矩阵乘法的数目大约极小化. Paterson and Stockmeyer(1973) 讨论了这一技巧. Van Loan(1978) 给出了不用存储 $\boldsymbol{A}^2, \cdots, \boldsymbol{A}^s$ 就能使这一方法实现的技巧.

### 11.2.5　计算矩阵的幂

一个矩阵的幂的问题值得特别注意. 假设要计算 $\boldsymbol{A}^{13}$, 注意到 $\boldsymbol{A}^4 = (\boldsymbol{A}^2)^2, \boldsymbol{A}^8 = (\boldsymbol{A}^4)^2, \boldsymbol{A}^{13} = \boldsymbol{A}^8\boldsymbol{A}^4\boldsymbol{A}$, 只需 5 次矩阵乘法就可完成这一运算. 对一般情况, 算法如下.

**算法 11.2.2 (二进制求幂法)**　给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和正整数 $s$, 本算法计算幂 $\boldsymbol{F} = \boldsymbol{A}^s$.

　　设 $s = \sum_{k=0}^{t} \beta_k 2^k$ 为 $s$ 的二次幂的展开, $\beta_t \neq 0$

　　$Z = A; q = 0$

　　**while**　$\beta_q = 0$

　　　　$Z = Z^2; q = q+1$

　　**end**

　　$F = Z$

　　**for**　$k = q+1 : t$

　　　　$Z = Z^2$

　　　　**if**　$\beta_k \neq 0$

　　　　　　$F = FZ$

　　　　**end**

　　**end**

这一算法最多需要 2floor[$\log_2(s)$] 次矩阵乘法. 若 $s$ 是 2 的幂, 则只需 $\log_2(s)$ 次矩阵乘法.

### 11.2.6 矩阵函数的积分

本节我们简略介绍矩阵函数的积分. 假设 $f(\boldsymbol{A}t)$ 对所有的 $t\in[a,b]$ 有定义, 我们希望计算

$$\boldsymbol{F}=\int_a^b f(\boldsymbol{A}t)\mathrm{d}t.$$

和 (11.1.1) 一样, 上式是对每一元素逐个积分.

一般的求积分规则对 $\boldsymbol{F}$ 也是适用的. 例如, 应用 Simpson 公式, 有

$$\boldsymbol{F}\approx\tilde{\boldsymbol{F}}=\frac{h}{3}\sum_{k=0}^{m}w_k f(\boldsymbol{A}(a+kh)), \tag{11.2.6}$$

其中 $m$ 为偶数, $h=(b-a)/m, w_k$ 的值如下:

$$w_k=\begin{cases}1, & k=0,m,\\ 4, & k \text{ 为奇数},\\ 2, & k \text{ 为偶数}, k\neq 0,m.\end{cases}$$

如果 $(\mathrm{d}^4/\mathrm{d}z^4)f(zt)=f^{(4)}(zt)$ 在 $t\in[a,b]$ 上连续, 且 $f^{(4)}(\boldsymbol{A}t)$ 在同一区间上有定义, 则可证

$$\tilde{\boldsymbol{F}}=\boldsymbol{F}+\boldsymbol{E},$$

其中

$$\|\boldsymbol{E}\|_2\leqslant\frac{nh^4(b-a)}{180}\max_{a\leqslant t\leqslant b}\|f^{(4)}(\boldsymbol{A}t)\|_2. \tag{11.2.7}$$

用 $f_{ij}$ 和 $e_{ij}$ 分别表示 $\boldsymbol{F}$ 和 $\boldsymbol{E}$ 的 $(i,j)$ 元. 在上述假设下, 应用 Simpson 公式的标准误差界可得

$$|e_{ij}|\leqslant\frac{h^4(b-a)}{180}\max_{a\leqslant t\leqslant b}|e_i^T f^{(4)}(\boldsymbol{A}t)e_j|.$$

因为 $\|\boldsymbol{E}\|_2\leqslant n\max|e_{ij}|$ 且

$$\max_{a\leqslant t\leqslant b}|e_i^T f^{(4)}(\boldsymbol{A}t)e_j|\leqslant\max_{a\leqslant t\leqslant b}\|f^{(4)}(\boldsymbol{A}t)\|_2,$$

所以 (11.2.7) 成立. 当然, 在 (11.2.6) 的实际应用中, 函数 $f(\boldsymbol{A}(a+kh))$ 一般要用逼近的方法来估计. 因此, 整体误差包括逼近 $f(\boldsymbol{A}(a+kh))$ 所带来的误差和 Simpson 公式本身所具有的误差.

## 习　题

**11.2.1** (a) 设 $\boldsymbol{G}=\lambda\boldsymbol{I}+\boldsymbol{E}$ 是 $p\times p$ Jordan 块, 其中 $\boldsymbol{E}=(\delta_{i,j-1})$, 证明:

$$(\lambda\boldsymbol{I}+\boldsymbol{E})^k=\sum_{j=0}^{\min\{p-1,k\}}\binom{k}{j}\lambda^{k-j}\boldsymbol{E}^j.$$

(b) 利用 (a) 和定理 11.1.1 证明定理 11.2.3.

**11.2.2**　证明 (11.2.2)

**11.2.3**　证明：若 $\|\boldsymbol{A}\|_2 < 1$, 则 $\ln(\boldsymbol{I}+\boldsymbol{A})$ 有定义，且不等式 $\|\ln(\boldsymbol{I}+\boldsymbol{A})\|_2 \leqslant \|\boldsymbol{A}\|_2/(1-\|\boldsymbol{A}\|_2)$ 成立.

**11.2.4**　已知 $\boldsymbol{A}$ 是一 $n \times n$ 对称正定矩阵. (a) 证明存在唯一的对称正定矩阵 $\boldsymbol{X}$, 使得 $\boldsymbol{A} = \boldsymbol{X}^2$. (b) 证明：若 $\boldsymbol{X}_0 = \boldsymbol{I}$, $\boldsymbol{X}_{k+1} = (\boldsymbol{X}_k + \boldsymbol{A}\boldsymbol{X}_k^{-1})/2$, 则 $\boldsymbol{X}_k \to \sqrt{\boldsymbol{A}}$, 这里平方根 $\sqrt{\boldsymbol{A}}$ 表示 (a) 中的 $\boldsymbol{X}$.

**11.2.5**　对矩阵 $\boldsymbol{A}$ 是对称的和上三角形的两种情况改写算法 11.2.1, 并给出相应的 flop 数.

**11.2.6**　证明：$\boldsymbol{X}(t) = C_1\cos(t\sqrt{\boldsymbol{A}}) + C_2\sqrt{\boldsymbol{A}^{-1}}\sin(t\sqrt{\boldsymbol{A}})$ 是初值问题 $\ddot{X}(t) = -\boldsymbol{A}\boldsymbol{X}(t)$, $X(0) = C_1, \dot{X}(0) = C_2$ 的解. 这里假定 $\boldsymbol{A}$ 是对称正定的.

**11.2.7**　利用定理 11.2.4 给出下列逼近的误差界：

$$\sin(\boldsymbol{A}) \approx \sum_{k=0}^{q}(-1)^k\frac{\boldsymbol{A}^{2k+1}}{(2k+1)!}, \quad \cos(\boldsymbol{A}) \approx \sum_{k=0}^{q}(-1)^k\frac{\boldsymbol{A}^{2k}}{(2k)!}$$

**11.2.8**　设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 非奇异, $\boldsymbol{X}_0 \in \mathbb{R}^{n\times n}$ 给定. 迭代

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k(2\boldsymbol{I} - \boldsymbol{A}\boldsymbol{X}_k)$$

是函数 $f(x) = a - (1/x)$ 的 Newton 法的矩阵形式. 应用 SVD 分析这一迭代. 它收敛到 $\boldsymbol{A}^{-1}$ 吗？讨论初值 $X_0$ 的选取.

## 本节注释与参考文献

多项式计算之 Horner's 方法的最优性可见：

D. Knuth(1981). *The Art of Computer Programming, vol.2. Seminumerical Algo-rithms,* 2nd ed., Addison-Wesley, Reading, Massachusetts.

M. S. Paterson and L. J. Stockmeyer(1973). "On the Number of Nonscalar Multiplications Necessary to Evaluate Polynomials," *SIAM J. Comp.* 2, 60–66.

关于矩阵多项式的 Horner 计算方式的分析可见：

C. F. Van Loan(1978). "A Note on the Evaluation of Matrix Polynomials," *IEEE Trans. Auto. Cont.* *AC*-24, 320-321.

矩阵函数计算的其他方面可见：

N. J. Higham and P. A. Knight(1995). "Matrix Powers in Finite Precision Arithmetic," *SIAM J. Matrix Anal. Appl.* 16,343-358.

R. Mathias(1993). "Approximation of Matrix-Valued Functions," *SIAM J. Matrix Anal. Appl.* 14, 1061–1063.

S. Friedland(1991). "Revisiting Matrix Squaring," *Lin. Alg. and Its Applic.*154–156, 59–63.

H. Bolz and W. Niethammer(1988). "On the Evaluation of Matrix Functions Given by Power Series," *SIAM J. Matrix Anal. Appl.* 9, 202–209.

$f(\boldsymbol{A})$ 的 Newton 表示和 Langrage 表示以及它们与其他矩阵函数定义的关系可见：

R. F. Rinehart(1955). "The Equivalence of Definitions of a Matric Function," *Amer. Math. Monthly.*62,395–414.

计算矩阵余弦值的“倍角”方法之分析可见:

S. Serbin and S. Blalock(1979). “An Algorithm for Computing the Matrix Cosine,” *SIAM J. Sci. Stat. Comp.* 1, 198–204.

平方根是一重要的矩阵函数, 见 4.2.10 节. 处理它有好几种途径:

Å. Björck and S.Hammarling(1983). “A Schur Method for the Square Root of a Matrix,” *Lin. Alg. and Its Applic.* 52/53, 127–140.

N.J. Higham(1986). “Newton's Method for the Matrix Square Root,” *Math. Comp.* 46, 537–550.

N.J. Higham(1987). “Computing Real Square Roots of a Real Matrix,” *Lin. Alg. and Its Applic,* 88/89, 405–430.

## 11.3 矩阵指数

经常要计算的矩阵函数之一是指数:

$$e^{\boldsymbol{A}t} = \sum_{k=0}^{\infty} \frac{(\boldsymbol{A}t)^k}{k!}.$$

计算它的算法已有许多, 但正如 Moler and Van Loan(1978) 的综述性文章所指出的那样, 它们中大多数在数值上都是不可靠的. 为了说明计算的困难所在, 我们在Padé逼近的基础上提出一种“加权与平方”的方法. 然后, 又给出了此方法的简要分析, 这里涉及 $e^{\boldsymbol{A}t}$ 的一些扰动理论. 在非正规矩阵的情形, 我们指出了特征分析的缺点.

### 11.3.1 Padé 逼近法

按照 11.2 节讨论的结果, 若 $g(z) \approx e^z$, 则 $g(\boldsymbol{A}) \approx e^{\boldsymbol{A}}$. 为达到这一目的, 有一类非常有用的逼近函数 ——Padé函数:

$$R_{pq}(z) = D_{pq}(z)^{-1} N_{pq}(z),$$

其中

$$N_{pq}(z) = \sum_{k=0}^{p} \frac{(p+q-k)!p!}{(p+q)!k!(p-k)!} z^k,$$

$$D_{pq}(z) = \sum_{k=0}^{q} \frac{(p+q-k)!q!}{(p+q)!k!(q-k)!} (-z)^k.$$

注意, $R_{po}(z) = 1 + z + \cdots + z^p/p!$ 是 $p$ 阶 Taylor 多项式.

不幸的是, Padé逼近仅在原点附近才非常好, 下面的等式说明了这一点:

$$e^{\boldsymbol{A}} = R_{pq}(\boldsymbol{A}) + \frac{(-1)^q}{(p+q)!} \boldsymbol{A}^{p+q+1} D_{pq}(\boldsymbol{A})^{-1} \int_0^1 u^p (1-u)^q e^{\boldsymbol{A}(1-u)} \mathrm{d}u. \qquad (11.3.1)$$

然而, 利用事实 $e^{\boldsymbol{A}} = (e^{\boldsymbol{A}/m})^m$ 可以来克服这一困难. 特别地, 我们可以把 $\boldsymbol{A}$ 除以 $m$, 使得 $F_{pq} = R_{pq}(\boldsymbol{A}/m)$ 是 $e^{\boldsymbol{A}/m}$ 比较精确的逼近. 然后用算法 11.2.2 计算 $F_{pq}^m$.

如果 $m$ 是 2 的幂, 则这相当于重复地平方, 因而非常有效. 整个算法的好坏取决于逼近

$$F_{pq} = \left(\boldsymbol{R}_{pq}\left(\frac{\boldsymbol{A}}{2^j}\right)\right)^{2^j}$$

的精度. Moler and Van Loan(1978) 证明了: 若 $\dfrac{||\boldsymbol{A}||_\infty}{2^j} \leqslant \dfrac{1}{2}$, 则存在矩阵 $\boldsymbol{E} \in \mathbb{R}^{n\times n}$, 使得

$$\begin{aligned}
&F_{pq} = \mathrm{e}^{\boldsymbol{A}+\boldsymbol{E}},\\
&\boldsymbol{AE} = \boldsymbol{EA},\\
&||\boldsymbol{E}||_\infty \leqslant \varepsilon(p,q)||\boldsymbol{A}||_\infty,\\
&\varepsilon(p,q) = 2^{3-(p+q)}\frac{p!q!}{(p+q)!(p+q+1)!}.
\end{aligned}$$

这些结果形成了有效计算 $\mathrm{e}^{\boldsymbol{A}}$ 且控制误差的方法之基础. 利用上面的公式, 易证不等式

$$\frac{||\mathrm{e}^{\boldsymbol{A}} - F_{pq}||_\infty}{||\mathrm{e}^{\boldsymbol{A}}||_\infty} \leqslant \varepsilon(p,q)||\boldsymbol{A}||_\infty \mathrm{e}^{\varepsilon(p,q)||\boldsymbol{A}||_\infty},$$

参数 $p, q$ 可以根据所要求的相对误差容限来确定. 注意, 既然 $F_{pq}$ 大约需要 $j + \max(p,q)$ 次矩阵乘法, 所以最好令 $p = q$, 因为这对给定的工作量, 使 $\varepsilon(p,q)$ 极小. 把这些思想结合起来, 就得到下面的算法.

**算法 11.3.1** 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 $\delta > 0$, 本算法计算出矩阵 $\boldsymbol{F} = \mathrm{e}^{\boldsymbol{A}+\boldsymbol{E}}$, 其中 $||\boldsymbol{E}||_\infty \leqslant \delta||\boldsymbol{A}||_\infty$.

$j = \max(0, 1 + \mathrm{floor}(\log_2(||A||_\infty)))$
$A = A/2^j$
令 $q$ 为满足 $\varepsilon(q,q) \leqslant \delta$ 的最小的非负整数
$D = I; N = I; X = I; c = 1$
**for** $k = 1:q$
  $c = c(q-k+1)/[(2q-k+1)k]$
  $X = AX; N = N + cX; D = D + (-1)^k cX$
**end**
用高斯消去法从 $DF = N$ 解出 $F$.
**for** $k = 1:j$
  $F = F^2$
**end**

这一算法需要大约 $2(q+j+1/3)n^3$ 个 flop. Ward(1977) 彻底分析了它的舍入误差性质.

11.2 节中特殊的 Horner 技巧可用来加快 $D = D_{qq}(\boldsymbol{A})$ 和 $N = N_{qq}(\boldsymbol{A})$ 的计算. 例如, 若 $q$=8, 则有 $N_{qq}(\boldsymbol{A}) = \boldsymbol{U} + \boldsymbol{AV}, D_{qq}(\boldsymbol{A}) = \boldsymbol{U} - \boldsymbol{AV}$, 其中

$$\boldsymbol{U} = c_0\boldsymbol{I} + c_2\boldsymbol{A}^2 + (c_4\boldsymbol{I} + c_6\boldsymbol{A}^2 + c_8\boldsymbol{A}^4)\boldsymbol{A}^4,$$

$$V = c_1 I + c_3 A^2 + (c_5 I + c_7 A^2) A^4.$$

显然, 只用 5 次矩阵乘法就可算出 $N$ 和 $D$, 但算法 11.3.1 却需要 7 次矩阵乘法.

### 11.3.2 扰动理论

有舍入误差时算法 11.3.1 是否稳定呢? 回答这一问题需要了解矩阵指数对矩阵 $A$ 中扰动的敏感度. 讨论的出发点是初值问题:

$$\dot{X}(t) = AX(t), \quad X(0) = I,$$

其中, $A, X(t) \in \mathbb{R}^{n\times n}$. 它有唯一解 $X(t) = e^{At}$, 矩阵指数的此特征可用来推导恒等式:

$$e^{(A+E)t} - e^{At} = \int_0^t e^{A(t-s)} E e^{(A+E)s} ds.$$

利用上式可得

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \leqslant \frac{\|E\|_2}{\|e^{At}\|_2} \int_0^t \|e^{A(t-s)}\|_2 \|e^{(A+E)s}\|_2 ds.$$

如果我们估计出被积函数中指数范数的界, 则结果可进一步简化. 这样做的一条途径是利用 Schur 分解. 若 $Q^{H}AQ = \text{diag}(\lambda_i) + N$ 是 $A \in \mathbb{C}^{n\times n}$ 的 Schur 分解, 则可证:

$$\|e^{At}\|_2 \leqslant e^{\alpha(A)tM_s(t)}, \tag{11.3.2}$$

其中

$$\alpha(A) = \max\{\text{Re}(\lambda) : \lambda \in \lambda(A)\}, \tag{11.3.3}$$

$$M_S(t) = \sum_{k=0}^{n-1} \frac{\|Nt\|_2^k}{k!}.$$

数 $\alpha(A)$ 称为谱的横坐标, 简单的运算可知:

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \leqslant t\|E\|_2 M_S(t)^2 \exp(tM_S(t)\|E\|_2).$$

注意, $M_S(t) \equiv 1$ 当且仅当 $A$ 是正规的, 这表明当 $A$ 是正规矩阵时, 矩阵指数 $e^{At}$ 是"良态的". 这一点被它的条件数所证实. **矩阵指数的条件数** $\nu(A,t)$ 定义如下:

$$\nu(A,t) = \max_{\|E\|_2 \leqslant 1} \left\| \int_0^t e^{A(t-s)} E e^{As} ds \right\|_2 \frac{\|A\|_2}{\|e^{At}\|_2}.$$

Van Loan(1972) 讨论了该数值衡量映射 $A \to e^{At}$ 的敏感度, 对给定的 $t$, 存在矩阵 $E$ 使得

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \approx \nu(A,t) \frac{\|E\|_2}{\|A\|_2}.$$

因此, 如果 $\nu(A,t)$ 很大, 则 $A$ 中很小的变化会导致 $e^{At}$ 中相对很大的变化. 不幸的是, 很难精确地刻画使 $\nu(A,t)$ 很大的矩阵 $A$(这和线性方程组 $Ax = b$ 相反,

那里病态的矩阵 $\boldsymbol{A}$ 可用奇异值分解 SVD 来简洁地描述). 然而, 可以肯定的是, $\nu(\boldsymbol{A},t) \geqslant t\|\boldsymbol{A}\|_2$, 等号对所有的非负 $t$ 成立当且仅当 $\boldsymbol{A}$ 是正规矩阵.

多注意一下非正规的影响, 就可从 11.2 节知道逼近 $e^{\boldsymbol{A}t}$ 所涉及的不仅仅是在 $\lambda(\boldsymbol{A})$ 上逼近 $e^{zt}$. 另外一条线索是由特征值不能得出问题 $e^{\boldsymbol{A}t}$ 的全部性态, 这与谱的横坐标 (11.3.3) 不能预测时间函数 $\|e^{\boldsymbol{A}t}\|_2$ 的大小是相关的. 如果 $\boldsymbol{A}$ 是正规矩阵, 则

$$\|e^{\boldsymbol{A}t}\|_2 = e^{\alpha(\boldsymbol{A})t}. \tag{11.3.4}$$

因此, 如果 $\boldsymbol{A}$ 的特征值都在左半开平面的话, 则上式是一致衰减的. 但若 $\boldsymbol{A}$ 是非正规矩阵, 则 $e^{\boldsymbol{A}t}$ 在衰减以前可能会增长. 下面的 $2 \times 2$ 矩阵

$$\boldsymbol{A} = \begin{bmatrix} -1 & M \\ 0 & -1 \end{bmatrix} \Leftrightarrow e^{\boldsymbol{A}t} = e^{-t}\begin{bmatrix} 1 & tM \\ 0 & 1 \end{bmatrix}$$

很清楚地说明了这一点.

### 11.3.3　一些稳定性问题

有了上面讨论我们可开始考虑算法 11.3.1 的稳定性. 如果 $\boldsymbol{A}$ 是一个指数在衰减前增长的矩阵, 在平方的过程中会出现一些潜在的困难. 如果

$$G = R_{qq}\left(\frac{\boldsymbol{A}}{2^j}\right) \approx e^{\boldsymbol{A}/2^j},$$

则可证舍入误差界的数量级为

$$\gamma = \mathbf{u}\|G^2\|_2\|G^4\|_2 \cdots \|G^{2^{j-1}}\|_2,$$

它损害所计算的 $G^{2^j}$ 是意料之中的. 如果 $\|e^{\boldsymbol{A}t}\|_2$ 的初始增长很大, 则可能

$$\gamma \gg \mathbf{u}\|G^{2^j}\|_2 \approx \mathbf{u}\|e^{\boldsymbol{A}}\|_2,$$

从而排除了相对误差较小的可能.

如果 $\boldsymbol{A}$ 是正规矩阵, 则 $\boldsymbol{G}$ 也是正规的, 因而对所有正整数 $m$ 有 $\|\boldsymbol{G}^m\|_2 = \|\boldsymbol{G}\|_2^m$. 所以, $\gamma \approx \mathbf{u}\|\boldsymbol{G}^{2^j}\|_2 \approx \mathbf{u}\|e^{\boldsymbol{A}}\|_2$, 故初始增长的问题消失了. 当 $\boldsymbol{A}$ 是正规矩阵时, 基本上能保证算法的相对误差较小. 反过来, 当 $\boldsymbol{A}$ 是非正规矩阵时, 要对方法给出结论更加困难, 因为还不清楚矩阵指数的条件数 $\nu(\boldsymbol{A},t)$ 与初始增长现象之间的联系. 然而, 数值实验表明, 只有当 $\nu(\boldsymbol{A},1)$ 很大时, 算法 11.3.1 不能给出相对精确的 $e^{\boldsymbol{A}}$.

### 11.3.4　特征值和伪特征值

在 7.1 节的结尾, 我们指出, 用矩阵的特征值来衡量与奇异矩阵的接近程度时往往不是很好, 除非矩阵是正规的. 在方程组 $Ax = b$ 的扰动分析中, 正是矩阵的奇异值反映了扰动的灵敏度. 我们对矩阵指数的讨论是对这种效应的另一个警示. 一个非正规矩阵 $\boldsymbol{A}$ 的谱不能完全描述 $e^{\boldsymbol{A}t}$ 的性态.

在许多应用中, 矩阵的特征值对所建模的基本现象“说明”些什么. 但当特征值对扰动极为敏感时, 它们所说明的则可能会是误导. 这促使了伪谱的思想的发展. 对 $\varepsilon > 0$, 矩阵 $\boldsymbol{A}$ 的 $\varepsilon$伪谱定义如下:

$$\lambda_\varepsilon(\boldsymbol{A}) = \left\{ z \in \mathbb{C} : \|(z\boldsymbol{I} - \boldsymbol{A})^{-1}\|_2 \geqslant \frac{1}{\varepsilon} \right\}. \tag{11.3.5}$$

定性地讲, 当 $z\boldsymbol{I}-\boldsymbol{A}$ 充分接近奇异矩阵时, $z$ 即是 $\boldsymbol{A}$ 的一个伪特征值. 约定 $\lambda_0(\boldsymbol{A}) = \lambda(\boldsymbol{A})$. 下面是伪谱的一些性质:

(1) 若 $\varepsilon_1 \leqslant \varepsilon_2$, 则 $\lambda_{\varepsilon_1}(\boldsymbol{A}) \subseteq \lambda_{\varepsilon_2}(\boldsymbol{A})$;

(2) $\lambda_\varepsilon(\boldsymbol{A}) = \{z \in \mathbb{C} : \sigma_{\min}(z\boldsymbol{I} - \boldsymbol{A}) \leqslant \varepsilon\}$;

(3) $\lambda_\varepsilon(\boldsymbol{A}) = \{z \in \mathbb{C} : z \in \lambda(\boldsymbol{A} + \boldsymbol{E}),\ \boldsymbol{E}$ 为满足 $\|\boldsymbol{E}\|_2 \leqslant \varepsilon$ 的矩阵$\}$.

描绘一非正规矩阵 $\boldsymbol{A}$ 的伪谱能反映出该矩阵的内在表现. 这里, “表现”可以指求解方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 的迭代法的数学性态, 也可指在涉及矩阵 $\boldsymbol{A}$ 的模型中所预测的物理现象. 见 Higham and Trefethen(1993), Nachtigal, Reddy, and Trefethen(1992) 及 Trefethen, Reddy, and Driscoll(1993).

## 习　题

**11.3.1** 证明: $e^{(\boldsymbol{A}+\boldsymbol{B})t} = e^{\boldsymbol{A}t}e^{\boldsymbol{B}t}$ 对所有的 $t$ 成立当且仅当 $\boldsymbol{AB} = \boldsymbol{BA}$(提示: 将两边展成 $t$ 的幂级数然后比较 $t$ 的系数).

**11.3.2** 设 $\boldsymbol{A}$ 是反对称的, 则 $e^{\boldsymbol{A}}$ 和 (1, 1)Padé逼近 $R_{11}(\boldsymbol{A})$ 都是正交的. 还有其他的 $p, q$ 使得 $R_{pq}(\boldsymbol{A})$ 正交吗?

**11.3.3** 证明: 若 $\boldsymbol{A}$ 非奇异, 则存在矩阵 $\boldsymbol{X}$ 使得 $\boldsymbol{A} = e^{\boldsymbol{X}}$, $\boldsymbol{X}$ 唯一吗?

**11.3.4** 证明: 若 $\exp\left(\begin{bmatrix} -\boldsymbol{A}^{\mathrm{T}} & \boldsymbol{P} \\ 0 & \boldsymbol{A} \end{bmatrix} z\right) = \underset{\begin{matrix} n & n \end{matrix}}{\begin{bmatrix} \boldsymbol{F}_{11} & \boldsymbol{F}_{12} \\ 0 & \boldsymbol{F}_{22} \end{bmatrix}}\begin{matrix} n \\ n \end{matrix}$, 则

$$\boldsymbol{F}_{11}^{\mathrm{T}}\boldsymbol{F}_{12} = \int_0^z e^{\boldsymbol{A}^{\mathrm{T}}t}\boldsymbol{P}e^{\boldsymbol{A}t}\mathrm{d}t.$$

**11.3.5** 当 $\boldsymbol{A} = \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}, \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ 时, 给出计算 $e^{\boldsymbol{A}}$ 的算法.

**11.3.6** 假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$且$\boldsymbol{v} \in \mathbb{R}^n, \|\boldsymbol{v}\|_2 = 1$, 定义函数 $\phi(t) = \|e^{\boldsymbol{A}t}\boldsymbol{v}\|_2^2/2$, 证明:

$$\phi(t) \leqslant \mu(\boldsymbol{A})\phi(t),$$

其中 $\mu(\boldsymbol{A}) = \lambda_1((\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}})/2)$. 由此可得结论:

$$\|e^{\boldsymbol{A}t}\|_2 \leqslant e^{\mu(\boldsymbol{A})t}, \quad t \geqslant 0.$$

**11.3.7** 证明本节中给出的三个关于伪谱的性质.

## 本节注释与参考文献

本节中出现的大部分内容以及大量的参考文献可见:

C. B. Moler and C. F. Van Loan(1978).“Nineteen Dubious Ways to Compute the Exponential of a Matrix,” *SIAM Review* 20, 801–836.

在仔细考查的 19 个方法中, 用 Padé逼近的加权与平方 (算法 11.3.1) 和 Darlett 的 Schur 分解方法 (算法 11.1.1) 的一个细致实现是相对可靠的方法. 关于矩阵指数 Padé 逼近的各个方面可参见:

W. Fair and Y. Luke(1970). "Padé Approximations to the Operator Exponential," *Numer. Math.* 14, 379-382.

C. F. Van Loan(1977). "On the Limitation and Application of Padé Approximation to the Matrix Exponential," in *Padé and Rational Approximation*, ed. E. B. Saff and R.S.Varga Academic Press, New York.

R. C. Ward(1977). "Numerical Computation of the Matrix Exponential with Accuracy Estimate," *SIAM J. Num. Anal.* 14, 600–614.

A. Wragg(1973). "Computation of the Exponential of a Matrix I: Theoretical Considerations," *J. inst. Math. Applic.* 11, 369–375.

A. Wragg(1975). "Computation of the Exponential of a Matrix II: Practical Considerations," *J. Inst. Math. Applic.* 15, 273–278.

方程 (11.3.1) 在标量情况下的证明可见:

R. S. Varga(1961). "On Higher-Order Stable Implicit Methods for Solving Parabolic Par-tial Differential Equations," *J.Math. Phys.* 40, 220–231.

在控制论的许多应用中, 要计算矩阵指数. 例如, 在线性最优调节问题中, 需要各种各样的涉及矩阵指数的积分, 见:

J. Johnson and C. L. Phillips(1971). "An Algorithm for the Computation of the Integral of the State Transition Matrix," *IEEE Trans. Auto, Cont.AC*-16, 204–205.

C. F. Van Loan(1978). "Computing Integrals Involving the Matrix Exponential," *IEEE Trans. Auto. Cont.AC*-23, 395–404.

在评估计算矩阵指数的算法的表现时，理解映射 $A \to \exp(At)$ 的敏感性是很有帮助的. 这方面的著作包括:

B. Kågström(1977). "Bounds and Perturbation Bounds for the Matrix Exponential," *BIT*17, 39–57.

C. F. Van Loan(1977). "The Sensitivity of the Matrix Exponential," *SIAM J. Num. Anal.* 14, 971–981.

R. Mathias(1992). "Evaluating the Frechet Derivative of the Matrix Exponential," *Numer. Math.* 63, 213–226.

矩阵对数的计算是一个重要的领域，它需要更多的研究. 这些计算出现在各种各样的系统识别问题中. 见:

B. Singer and S. Spilerman(1976). "The Representation of Social Processes by Markov Models," *Amer. J. Sociology* 82, 1–54.

B. W. Helton(1968). "Logarithms of Matrices ," *Proc. Amer. Math. Soc.* 19, 733–736.

关于伪谱，我们建议参阅：

L. N. Trefethen(1992). "Pseudospecta of Matrices," in *Numerical Analysis* 19[illegible] D. F. Griffiths and G. A. Watson(eds), Longman Scientific and Technical, Harlow , Essex, UK, 234–262.

D. J. Higham and L. N. Trefethen(1993). "Stiffness of ODES," *BIT* 33, 285–303.

L. N. Trefethen, A. E. Trefethen, S. C. Reddy, and T. A. Driscoll(1993). "Hydrodynamic Stability Without Eigenvalues," *Science* 261, 578–584.

以及 Chaitin-Chatelin and Frayssé(1996, 第 10 章).

# 第12章　特殊问题

本章我们讨论几类反映奇异值、QR 分解和 Schur 分解重要应用的问题. 首先考虑带约束的最小二乘极小化问题. 12.1 节阐述了两类约束: 二次不等式和线性等式. 紧接着的两节讨论标准最小二乘问题的变化形式. 在 12.2 节中, 我们讨论了怎样用矩阵 $\boldsymbol{A}$ 的某些列来逼近观察向量 $\boldsymbol{b}$, 当 $\boldsymbol{A}$ 秩亏时, 这是经常用到的. 在 12.3 节中, 我们考虑了被称为整体最小二乘的一般回归问题的变化形式, 它在 $\boldsymbol{A}$ 有误差时是很有用的. 有关奇异值分解更多的应用在 12.4 节讨论, 其中考虑了几类子空间的计算问题. 12.5 节讨论了当矩阵 $\boldsymbol{A}$ 有低秩扰动时, 怎样更新它的正交分解. 基本特征问题的某些变化形式将在 12.6 节中讨论.

**预备知识**

本章是综述性的, 所以没有给出所需知识的章节范围. 而是在每节的开头都会提一下本书前面相关的内容, 且如果合适的话, 会提到 LAPACK 或其他专著.

## 12.1　约束最小二乘问题

在最小二乘中, 有时很自然地要在 $\mathbb{R}^n$ 的某子集上极小化 $\|\boldsymbol{Ax}-\boldsymbol{b}\|_2$. 例如, 我们希望在单位球面 $\|\boldsymbol{x}\|_2=1$ 上用 $\boldsymbol{Ax}$ 来最佳预测向量 $\boldsymbol{b}$. 或者需要求出一个拟合函数使其在有限个点上取给定的值. 这就导致了等式约束的最小二乘问题, 本节讨论怎样用 QR 分解和奇异值分解来解决这些问题.

阅读本节需要第 5 章及 8.7 节的知识. 相应的 LAPACK 软件有:

| LAPACK: 解广义/约束最小二乘问题 | |
|---|---|
| _GGLSE | 解等式约束的 LS 问题 |
| _GGQRF | 计算矩阵对的广义 QR 分解 |
| _GGRQF | 计算矩阵对的广义 RQ 分解 |
| _GGSVP | 把广义奇异值分解问题转化为三角形形式 |
| _TGSJA | 计算一对三角形矩阵的广义奇异值分解 |

补充参考文献包括 Lawson and Hanson(1974) 以及 Björck(1996).

### 12.1.1　二次不等式约束最小二乘

带二次不等式约束的最小二乘的极小化, 简称 LSQI 问题, 是一种当一般的 LS 问题之解需要规范化时可用到的技术. 在对有噪声的数据进行插值时, 会出现下面简单的 LSQI 问题:

$$\min \|\boldsymbol{Ax}-\boldsymbol{b}\|_2, \text{在约束 } \|\boldsymbol{Bx}\|_2 \leqslant \alpha \text{ 下}, \tag{12.1.1}$$

其中 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{B} \in \mathbb{R}^{n\times n}$(非奇异) 且 $\alpha \geqslant 0$. 约束条件定义了 $\mathbb{R}^n$ 中的一超椭球体, 它用来阻尼拟合函数中过分的振荡. 例如, 当 $\boldsymbol{B}$ 是离散的二阶求导算子就可以做到这一点.

更一般地, 我们会遇到如下问题:

$$\min \|\boldsymbol{Ax}-\boldsymbol{b}\|_2, \text{在约束 } \|\boldsymbol{Bx}-\boldsymbol{d}\|_2 \leqslant \alpha \text{ 下}, \tag{12.1.2}$$

其中 $\boldsymbol{A} \in \mathbb{R}^{m\times n}(m \geqslant n), \boldsymbol{b} \in \mathbb{R}^m, \boldsymbol{B} \in \mathbb{R}^{p\times n}, \boldsymbol{d} \in \mathbb{R}^p$ 以及 $\alpha \geqslant 0$, 8.7.3 节中的广义奇异值分解为解 (12.1.2) 提供了思路. 事实上, 若

$$\begin{aligned} \boldsymbol{U}^{\mathrm{T}}\boldsymbol{AX} &= \operatorname{diag}(\alpha_1, \alpha_2, \cdots, \alpha_n), \quad \boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{I}_m, \\ \boldsymbol{V}^{\mathrm{T}}\boldsymbol{BX} &= \operatorname{diag}(\beta_1, \beta_2, \cdots, \beta_q), \quad \boldsymbol{V}^{\mathrm{T}}\boldsymbol{V} = \boldsymbol{I}_p,\ q = \min\{p, n\} \end{aligned} \tag{12.1.3}$$

是 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的广义奇异值分解, 则 (12.1.2) 变为

$$\min \|\boldsymbol{D_A y}-\tilde{\boldsymbol{b}}\|_2, \text{在约束 } \|\boldsymbol{D_B y}-\tilde{\boldsymbol{d}}\,\|_2 \leqslant \alpha \text{ 下},$$

其中 $\tilde{\boldsymbol{b}} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{b}, \tilde{\boldsymbol{d}} = \boldsymbol{V}^{\mathrm{T}}\boldsymbol{d}, \boldsymbol{y} = \boldsymbol{X}^{-1}\boldsymbol{x}$. 简单形式的目标函数

$$\|\boldsymbol{D_A y}-\tilde{\boldsymbol{b}}\|_2^2 = \sum_{i=1}^{n}(\alpha_i y_i - \tilde{b}_i)^2 + \sum_{i=n+1}^{m} \tilde{b}_i^2 \tag{12.1.4}$$

以及约束方程

$$\|\boldsymbol{D_B y}-\tilde{\boldsymbol{d}}\,\|_2^2 = \sum_{i=1}^{r}(\beta_i y_i - \tilde{d}_i)^2 + \sum_{i=r+1}^{p} \tilde{d}_i^2 \leqslant \alpha^2 \tag{12.1.5}$$

方便了 LSQI 问题的分析. 这里 $r = \operatorname{rank}(\boldsymbol{B})$, 且假定 $\beta_{r+1} = \cdots = \beta_q = 0$.

首先, 原问题有解当且仅当

$$\sum_{i=r+1}^{p} \tilde{d}_i^2 \leqslant \alpha^2.$$

如果在表达式中等号成立, 由 (12.1.4) 和 (12.1.5) 知向量 $y$:

$$y_i = \begin{cases} \tilde{d}_i/\beta_i, & i = 1:r, \\ \tilde{b}_i/\alpha_i, & i = r+1:n, \quad a_i \neq 0, \\ 0, & i = r+1:n, \quad \alpha_i = 0, \end{cases} \tag{12.1.6}$$

是原 LSQI 问题的解. 否则

$$\sum_{i=r+1}^{p} \tilde{d}_i^2 < \alpha^2, \tag{12.1.7}$$

我们有更多的选择. $\mathbb{R}^n$ 中的向量 $\boldsymbol{y}$:

$$y_i = \begin{cases} \tilde{b}_i/\alpha_i, & \alpha_i \neq 0, \\ \tilde{d}_i/\beta_i, & \alpha_i = 0, \end{cases} \quad i = 1:n$$

是 $\|D_A y-\tilde{b}\|_2$ 的极小点. 如果此向量也是可行的, 则得到 (12.1.2) 的一个解 (然而, 这并不一定是极小 2 范数解). 因此, 我们假设

$$\sum_{\substack{i=1\\ \alpha_i\neq 0}}^{q}\left(\beta_i\frac{\tilde{b}_i}{\alpha_i}-\tilde{d}_i\right)^2+\sum_{i=q+1}^{p}\tilde{d}_i^{\,2}>\alpha^2. \tag{12.1.8}$$

这意味着 LSQI 问题的解出现在可行域的边界上. 因此, 剩下的问题是

$$\min\|D_A y-\tilde{b}\|_2,\text{在约束 }\|D_B y-\tilde{d}\|_2=\alpha\text{ 下}.$$

现用 Lagrange 乘子法求解这一问题. 定义函数

$$h(\lambda,y)=\|D_A y-\tilde{b}\|_2^2+\lambda(\|D_B y-\tilde{d}\|_2^2-\alpha^2),$$

由 $0=\partial h/\partial y_i, i=1:n$, 得方程组

$$(D_A^{\mathrm{T}}D_A+\lambda D_B^{\mathrm{T}}D_B)y=D_A^{\mathrm{T}}\tilde{b}+\lambda D_B^{\mathrm{T}}\tilde{d}\,.$$

假设系数矩阵非奇异，则得解 $y(\lambda)$:

$$y_i(\lambda)=\begin{cases}\dfrac{\alpha_i\tilde{b}_i+\lambda\beta_i\tilde{d}_i}{\alpha_i^2+\lambda\beta_i^2}, & i=1:q,\\ \tilde{b}_i/\alpha_i, & i=q+1:n.\end{cases}$$

为了确定 Lagrange 参数, 定义函数

$$\phi(\lambda)\equiv\|D_B y(\lambda)-\tilde{d}\|_2^2=\sum_{i=1}^{r}\left(\alpha_i\frac{\beta_i\tilde{b}_i-\alpha_i\tilde{d}_i}{\alpha_i^2+\lambda\beta_i^2}\right)^2+\sum_{i=r+1}^{p}\tilde{d}_i^{\,2},$$

则由方程 $\phi(\lambda)=\alpha^2$ 确定 $\lambda$. 这一类型的方程称为特征 (secular) 方程, 它在 8.5.3 节中出现过. 从 (12.1.8) 可知, $\phi(0)>\alpha^2$. 当 $\lambda>0$ 时, $\phi(\lambda)$ 是单调减的, 因此存在唯一的正数 $\lambda^*$ 使得 $\phi(\lambda^*)=\alpha^2$, 易证这就是所要的根. 它可以用任何一种标准的求根方法来求, 例如 Newton 法. 原 LSQI 问题的解为 $x=Xy(\lambda^*)$.

### 12.1.2 球上的 LS 极小化

对球 $(B=I_n, d=0)$ 上的极小化的重要情形, 我们有下列算法.

**算法 12.1.1** 给定矩阵 $A\in\mathbb{R}^{m\times n}, m\geqslant n, b\in\mathbb{R}^m, \alpha>0$, 本算法计算一向量 $x\in\mathbb{R}^n$, 使得 $\|Ax-b\|_2$ 极小且满足约束 $\|x\|_2\leqslant\alpha$.

计算奇异值分解 $A=U\Sigma V^{\mathrm{T}}$, 存储矩阵 $V=[v_1,\cdots,v_n]$ 并生成向量 $b=U^{\mathrm{T}}b$.

$r=\text{rank}(A)$

**if** $\displaystyle\sum_{i=1}^{r}\left(\frac{b_i}{\sigma_i}\right)^2>\alpha^2$

求 $\lambda^*$ 使得 $\displaystyle\sum_{i=1}^{r}\left(\frac{\sigma_i b_i}{\sigma_i^2+\lambda^*}\right)^2=\alpha^2.$

$$x = \sum_{i=1}^{r}\left(\frac{\sigma_i b_i}{\sigma_i^2+\lambda^*}\right)v_i$$

**else**

$$x = \sum_{i=1}^{r}\left(\frac{b_i}{\sigma_i}\right)v_i$$

**end**

在此算法中，主要的计算是奇异值分解.

**例 12.1.1** LSQI 问题

$$\min_{\|\boldsymbol{x}\|_2=1}\left\|\begin{bmatrix}2&0\\0&1\\0&0\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}-\begin{bmatrix}4\\2\\3\end{bmatrix}\right\|_2$$

的特征方程为

$$\left(\frac{8}{\lambda+4}\right)^2+\left(\frac{2}{\lambda+1}\right)^2=1.$$

对这个问题, 解为 $\lambda^*=4.571\ 32$ 与 $\boldsymbol{x}=(0.933\ 4, 0.358\ 98)^{\mathrm{T}}$.

### 12.1.3 岭回归

算法 12.1.1 所求解的问题等价于 Lagrange 乘子问题：确定 $\lambda>0$ 使得

$$(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})\boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}, \tag{12.1.9}$$

且 $\|\boldsymbol{x}\|_2=\alpha$. 这正好是岭回归问题

$$\min_{\boldsymbol{x}}\left\|\begin{bmatrix}\boldsymbol{A}\\\sqrt{\lambda}\boldsymbol{I}\end{bmatrix}\boldsymbol{x}-\begin{bmatrix}\boldsymbol{b}\\0\end{bmatrix}\right\|_2^2=\min_{\boldsymbol{x}}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\lambda\|\boldsymbol{x}\|_2^2$$

的正规方程. 在一般的岭回归问题中, 用某种准则来选取岭参数 $\lambda$, 例如: 对某个给定的 $\alpha$, $\|\boldsymbol{x}(\lambda)\|_2=\alpha$. 我们介绍一下 Golub, Heath, and Wahba (1979) 的一个选取 $\lambda$ 的算法.

令 $\boldsymbol{D}_k=\boldsymbol{I}-\boldsymbol{e}_k\boldsymbol{e}_k^{\mathrm{T}}=\mathrm{diag}(1,\cdots,1,0,1,\cdots,1)\in\mathbb{R}^{m\times m}$, 设 $\boldsymbol{x}_k(\lambda)$ 为

$$\min_{\boldsymbol{x}}\|\boldsymbol{D}_k(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})\|_2^2+\lambda\|\boldsymbol{x}\|_2^2 \tag{12.1.10}$$

的解. 因此, $\boldsymbol{x}_k(\lambda)$ 是去掉矩阵 $\boldsymbol{A}$ 的第 $k$ 行和向量 $\boldsymbol{b}$ 的第 $k$ 个分量的岭回归问题的解, 即忽略了第 $k$ 次实验. 现考虑选择 $\lambda$, 使得交叉确认加权平方误差 $C(\lambda)$ 达到极小, 其中

$$C(\lambda)=\frac{1}{m}\sum_{k=1}^{m}w_k(\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{x}_k(\lambda)-\boldsymbol{b}_k)^2,$$

这里 $w_1,\cdots,w_m$ 是非负的加权数, $\boldsymbol{a}_k^{\mathrm{T}}$ 是 $\boldsymbol{A}$ 的第 $k$ 行. 注意到

$$\|\boldsymbol{A}\boldsymbol{x}_k(\lambda)-\boldsymbol{b}\|_2^2=\|\boldsymbol{D}_k(\boldsymbol{A}\boldsymbol{x}_k(\lambda)-\boldsymbol{b})\|_2^2+(\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{x}_k(\lambda)-\boldsymbol{b}_k)^2,$$

我们可以看出, 当 $\boldsymbol{A}$ 的第 $k$ 行又重新考虑时, 在原平方和中增加了 $(\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{x}_k(\lambda)-\boldsymbol{b}_k)^2$ 一项. 极小化 $\boldsymbol{C}(\lambda)$ 相当于选取 $\lambda$, 使得最终的模型不过分依赖于任何一次实验.

为了使上面的叙述更加精确, 我们对它进行严格的分析, 并给出一个极小化 $\boldsymbol{C}(\lambda)$ 的方法. 假设 $\lambda>0$, 通过简单的代数运算可得

$$\boldsymbol{x}_k(\lambda)=\boldsymbol{x}(\lambda)+\frac{\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{x}(\lambda)-\boldsymbol{b}_k}{1-\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{a}_k}\boldsymbol{z}_k, \tag{12.1.11}$$

其中 $\boldsymbol{z}_k=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})^{-1}\boldsymbol{a}_k, \boldsymbol{x}(\lambda)=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$, 在 (12.1.11) 的两边左乘 $-\boldsymbol{a}_k^{\mathrm{T}}$ 并都加上 $\boldsymbol{b}_k$, 则得到

$$\boldsymbol{b}_k-\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{x}_k(\lambda)=\frac{\boldsymbol{e}_k^{\mathrm{T}}(\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^{\mathrm{T}})\boldsymbol{b}}{\boldsymbol{e}_k^{\mathrm{T}}(\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^{\mathrm{T}})\boldsymbol{e}_k}. \tag{12.1.12}$$

注意到残差 $\boldsymbol{r}=(r_1,\cdots,r_m)^{\mathrm{T}}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}(\lambda)$ 由公式 $\boldsymbol{r}=[\boldsymbol{I}-\boldsymbol{A}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})^{-1}\boldsymbol{A}^{\mathrm{T}}]\boldsymbol{b}$ 给出，可知

$$\boldsymbol{C}(\lambda)=\frac{1}{m}\sum_{k=1}^{m}w_k\left(\frac{r_k}{\partial r_k/\partial b_k}\right)^2.$$

商 $r_k/(\partial r_k/\partial b_k)$ 可以看成是第 $k$ 次观测值 $b_k$ 对模型影响的反度量. 当 $\partial r_k/\partial b_k$ 很小时, 这表明模型预测 $b_k$ 的误差有点与 $b_k$ 无关. 选取参数 $\lambda^*$ 使得 $C(\lambda)$ 达到极小就是为了减小这种趋势.

计算 $\boldsymbol{A}$ 的奇异值分解后, $\lambda^*$ 的确定变得非常简单. 事实上, 若 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}=$ diag $(\sigma_1,\cdots,\sigma_n),\sigma_1\geqslant\cdots\geqslant\sigma_n,\tilde{\boldsymbol{b}}=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{b}$, 则由 (12.1.12) 可证

$$\boldsymbol{C}(\lambda)=\frac{1}{m}\sum_{k=1}^{m}w_k\left[\frac{\tilde{b}_k-\displaystyle\sum_{j=1}^{r}u_{kj}\tilde{b}_j\left(\frac{\sigma_j^2}{\sigma_j^2+\lambda}\right)}{1-\displaystyle\sum_{j=1}^{r}u_{kj}^2\left(\frac{\sigma_j^2}{\sigma_j^2+\lambda}\right)}\right]^2.$$

Golub, Heath, and Wahba(1979) 讨论了这个表达式的极小化问题.

### 12.1.4 等式约束的最小二乘

在本节的最后, 我们考虑约束条件为线性等式的最小二乘问题:

$$\min\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2, \text{在约束 } \boldsymbol{B}\boldsymbol{x}=\boldsymbol{d} \text{ 下}, \tag{12.1.13}$$

这里 $\boldsymbol{A}\in\mathbb{R}^{m\times n},\boldsymbol{B}\in\mathbb{R}^{p\times n},\boldsymbol{b}\in\mathbb{R}^m,\boldsymbol{d}\in\mathbb{R}^p$ 且 $\operatorname{rank}(\boldsymbol{B})=p$. 我们把 (12.1.13) 称为 LSE 问题. 在 (12.1.2) 中令 $\alpha=0$, 即得上面的问题 (11.1.13). 可见 LSE 问题是 LSQI 问题的一个特例. 然而, 直接求解 LSE 问题比 Lagrange 乘子法更为简单.

为简单起见, 假设 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是满秩的. 设

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}=\begin{bmatrix}\boldsymbol{R}\\\boldsymbol{0}\end{bmatrix}\begin{matrix}p\\n-p\end{matrix}$$

为 $\boldsymbol{B}^{\mathrm{T}}$ 的 QR 分解, 且作划分:

$$\boldsymbol{AQ} = [\ \underset{p}{\boldsymbol{A}_1} \quad \underset{n-p}{\boldsymbol{A}_2}\ ], \quad \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} \begin{matrix} p \\ n-p \end{matrix}.$$

很明显, 在这些变换下, 问题 (12.1.13) 变为

$$\min_{\boldsymbol{R}^{\mathrm{T}}\boldsymbol{y}=\boldsymbol{d}} \|\boldsymbol{A}_1\boldsymbol{y} + \boldsymbol{A}_2\boldsymbol{z} - \boldsymbol{b}\|_2.$$

因此, $\boldsymbol{y}$ 可从约束方程 $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{d}$ 解出, 向量 $\boldsymbol{z}$ 可以通过求解下面的无约束 LS 问题得出:

$$\min_{\boldsymbol{z}} \|\boldsymbol{A}_2\boldsymbol{z} - (\boldsymbol{b} - \boldsymbol{A}_1\boldsymbol{y})\|_2.$$

综上所述, 向量 $\boldsymbol{x} = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix}$ 为问题 (12.1.13) 的解.

**算法 12.1.2** *设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{B} \in \mathbb{R}^{p\times n}$, $\boldsymbol{b} \in \mathbb{R}^m, \boldsymbol{d} \in \mathbb{R}^p$. 如果 $\mathrm{rank}(\boldsymbol{A}) = n$ 且 $\mathrm{rank}(\boldsymbol{B}) = p$, 则本算法在约束条件 $\boldsymbol{Bx} = \boldsymbol{b}$ 下极小化 $\|\boldsymbol{Ax} - \boldsymbol{b}\|_2$.*

$B^{\mathrm{T}} = QR$ (QR 分解)

解方程 $R(1:p,1:p)^{\mathrm{T}}y = d$ 得到 $y$.

$A = AQ$

求 $Z$ 使得 $\|A(:,p+1:n)z - (b - A(:,1:p)y)\|_2$ 极小

$x = Q(:,1:p)y + Q(:,p+1:n)z$

注意, 这一算法用到了两次矩阵的分解和一次矩阵乘法.

### 12.1.5 加权法

一种求问题 (12.1.13) 近似解的有趣方法是对充分大的 $\lambda$ 解无约束 LS 问题

$$\min_{\boldsymbol{x}} \left\| \begin{bmatrix} \boldsymbol{A} \\ \lambda\boldsymbol{B} \end{bmatrix} \boldsymbol{x} - \begin{bmatrix} \boldsymbol{b} \\ \lambda\boldsymbol{d} \end{bmatrix} \right\|_2. \tag{12.1.14}$$

8.7.3 节中的广义奇异值分解 (GSVD) 可用来分析逼近的效果. 令

$$\begin{aligned} \boldsymbol{U}^{\mathrm{T}}\boldsymbol{AX} &= \mathrm{diag}(\alpha_1, \alpha_2, \cdots, \alpha_n) = \boldsymbol{D}_A \in \mathbb{R}^{m\times n}, \\ \boldsymbol{V}^{\mathrm{T}}\boldsymbol{BX} &= \mathrm{diag}(\beta_1, \beta_2, \cdots, \beta_p) = \boldsymbol{D}_B \in \mathbb{R}^{p\times n} \end{aligned}$$

为 $(\boldsymbol{A}, \boldsymbol{B})$ 的 GSVD. 为明确起见, 假设两矩阵都是满秩的. 若 $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m]$, $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_p], \boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, 则易证

$$\boldsymbol{x} = \sum_{i=1}^{p} \frac{\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{d}}{\beta_i} x_i + \sum_{i=p+1}^{n} \frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\alpha_i} x_i, \tag{12.1.15}$$

这正好是 (12.1.13) 的解, 而

$$x(\lambda) = \sum_{i=1}^{p} \frac{\alpha_i\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b} + \lambda^2\beta_i^2\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{d}}{\alpha_i^2 + \lambda^2\beta_i^2} x_i + \sum_{i=p+1}^{n} \frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\alpha_i} x_i \tag{12.1.16}$$

是 (12.1.14) 的解. 由

$$x(\lambda)-x=\sum_{i=1}^{p}\frac{\alpha_i(\beta_i\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}-\alpha_i\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{d})}{\beta_i(\alpha_i^2+\lambda^2\beta_i^2)}x_i \tag{12.1.17}$$

知当 $\lambda\to\infty$ 时 $x(\lambda)\to x$.

这种解 LSE 问题的好处是它不需要特殊的子程序：一般的 LS 求解算法即可. 然而, 当 $\lambda$ 的值很大时, 会遇到数值上的困难, 故有必要采用预防措施, 见 Powell and Reid(1968) 以及 Van Loan(1982a).

**例 12.1.2** 问题

$$\min_{x_1=x_2}\left\|\begin{bmatrix}1&2\\3&4\\5&6\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}-\begin{bmatrix}7\\1\\3\end{bmatrix}\right\|_2$$

的解为 $\boldsymbol{x}=[0.340\ 782\ 1,0.340\ 782\ 1]^{\mathrm{T}}$. 近似地可求解问题

$$\min\left\|\begin{bmatrix}1&2\\3&4\\5&6\\1000&-1000\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}-\begin{bmatrix}7\\1\\3\\0\end{bmatrix}\right\|_2,$$

其解为 $\boldsymbol{x}=[0.340\ 781\ 0,0.340\ 782\ 9]^{\mathrm{T}}$.

## 习 题

**12.1.1** (a) 证明: 若 $\mathrm{null}(\boldsymbol{A})\cap\mathrm{null}(\boldsymbol{B})\neq\{0\}$, 则 (12.1.2) 不可能有唯一解. (b) 给出一例子, 说明反过来是不对的 (提示: $\boldsymbol{A}^{+}\boldsymbol{b}$ 可行).

**12.1.2** 设 $p_0(x),p_1(x),\cdots,p_n(x)$ 是已知的多项式, $(x_0,y_0),\cdots,(x_m,y_m)$ 是一列已知的坐标对, $x_i\in[a,b]$. 我们希望找出一多项式 $p(x)=\sum_{k=0}^{n}a_kp_k(x)$ 使得 $\sum_{i=0}^{m}(p(x_i)-y_i)^2$ 在下列约束下达到极小:

$$\int_a^b[p''(x)]^2\mathrm{d}x\approx h\sum_{i=0}^{N}\left(\frac{p(z_{i-1})-2p(z_i)+p(z_{i+1})}{h^2}\right)^2\leqslant\alpha^2,$$

其中 $z_i=a+ih,b=a+Nh$, 证明: 这导致形如 (12.1.1) 的 LSQI 问题.

**12.1.3** 设 $\boldsymbol{Y}=[y_1,\cdots,y_k]\in\mathbb{R}^{m\times k}$ 满足

$$\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Y}=\mathrm{diag}(d_1^2,\cdots,d_k^2),\quad d_1\geqslant d_2\geqslant\cdots\geqslant d_k>0.$$

证明: 若 $\boldsymbol{Y}=\boldsymbol{Q}\boldsymbol{R}$ 为 $\boldsymbol{Y}$ 的 QR 分解, 则 $\boldsymbol{R}$ 是对角矩阵且 $|r_{ii}|=d_i$.

**12.1.4** (a) 证明: 若 $(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})\boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b},\lambda>0,\|\boldsymbol{x}\|_2=\alpha$, 则 $\boldsymbol{z}=(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})/\lambda$ 是对偶方程 $(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}+\lambda\boldsymbol{I})\boldsymbol{z}=-\boldsymbol{b},\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{z}\|_2=\alpha$ 的解. (b) 证明: 若 $(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}+\lambda\boldsymbol{I})\boldsymbol{z}=-\boldsymbol{b},\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{z}\|_2=\alpha$, 则 $\boldsymbol{x}=-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{z}$ 满足 $(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\lambda\boldsymbol{I})\boldsymbol{x}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b},\|\boldsymbol{x}\|_2=\alpha$.

**12.1.5** 假设 $\boldsymbol{A}$ 是元素均为 1 的 $m\times 1$ 矩阵, $\boldsymbol{b}\in\mathbb{R}^m$, 证明: 用单位加权的交叉确认技巧给出了最优值 $\lambda$,

$$\lambda=\left(\left(\frac{\tilde{b}}{s}\right)^2-\frac{1}{m}\right)^{-1},$$

其中 $\tilde{b}^{\mathrm{T}}=(b_1+\cdots+b_m)/m, s=\sum_{i=1}^m(b_i-\tilde{b})^2/(m-1)$.

**12.1.6** 证明方程 (12.1.15), (12.1.16) 和 (12.1.17).

**12.1.7** 给出算法 12.1.2 的 SVD 形式, 使之能处理秩亏损的 $\boldsymbol{A}$ 和 $\boldsymbol{B}$.

**12.1.8** 设 $\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1\\ \boldsymbol{A}_2\end{bmatrix}$, 其中 $\boldsymbol{A}_1\in\mathbb{R}^{n\times n}$ 非奇异, $\boldsymbol{A}_2\in\mathbb{R}^{(m-n)\times n}$. 证明:

$$\sigma_{\min}(\boldsymbol{A})\geqslant\sqrt{1+\sigma_{\min}(\boldsymbol{A}_2\boldsymbol{A}_1^{-1})^2}\,\sigma_{\min}(\boldsymbol{A}_1).$$

**12.1.9** 考虑问题

$$\min_{\substack{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{x}=\beta^2\\ \boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{x}=\gamma^2}}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2,\quad \boldsymbol{A}\in\mathbb{R}^{m\times n},\boldsymbol{b}\in\mathbb{R}^m,\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{n\times n}$$

假设 $\boldsymbol{B}$ 和 $\boldsymbol{C}$ 是正定的, 且 $\boldsymbol{Z}\in\mathbb{R}^{n\times n}$ 为非奇异矩阵, 满足 $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{Z}=\mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_n)$, $\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{Z}=\boldsymbol{I}_n,\lambda_1\geqslant\cdots\geqslant\lambda_n$, (a) 证明: 除非 $\lambda_n\leqslant\beta^2/\gamma^2\leqslant\lambda_1, x$ 的可行集是空的. (b) 利用 $\boldsymbol{Z}$, 说明如何把两个约束的问题转化成一个约束的问题:

$$\min_{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{y}=\beta^2-\lambda_n\gamma^2}\|\tilde{\boldsymbol{A}}\,\boldsymbol{x}-\boldsymbol{b}\|_2,$$

其中 $\boldsymbol{W}=\mathrm{diag}(\lambda_1,\cdots,\lambda_n)-\lambda_n\boldsymbol{I}$.

**12.1.10** 设 $p\geqslant m\geqslant n$, 且 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$, $\boldsymbol{B}\in\mathbb{R}^{m\times p}$, 说明如何计算正交矩阵 $\boldsymbol{Q}\in\mathbb{R}^{m\times m}$ 和 $\boldsymbol{V}\in\mathbb{R}^{n\times n}$ 使得

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}\boldsymbol{R}\\ 0\end{bmatrix},\quad \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{V}=[0,\boldsymbol{S}],$$

其中 $\boldsymbol{R}\in\mathbb{R}^{n\times n}$ 和 $\boldsymbol{S}\in\mathbb{R}^{m\times m}$ 都是上三角矩阵.

**12.1.11** 设 $\boldsymbol{r}\in\mathbb{R}^m$, $\boldsymbol{y}\in\mathbb{R}^n$, $\delta>0$, 阐明如何求下述问题:

$$\min_{\substack{\boldsymbol{E}\in\mathbb{R}^{m\times n}\\ \|\boldsymbol{E}\|_F\leqslant\delta}}\|\boldsymbol{E}\boldsymbol{y}-\boldsymbol{r}\|_2.$$

把 “min” 换成 “max”, 情况又如何?

## 本节注释与参考文献

粗略地说, 正规化是把坏条件问题转换成稳定问题的一种技巧. 二次约束的最小二乘问题是一重要的例子, 见:

L. Eldén(1977). “Algorithms for the Regularization of Ill-Conditioned Least Squares Problems,” *BIT* 17, 134–145.

交叉确认的文献有:

G. H. Golub, M. Heath, and G. Wahba(1979). “Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter,” *Technometrics* 21, 215–223.

L. Eldén(1985). “A Note on the Computation of the Generalized Cross-Validation Function for Ill-Conditioned Least Squares Problems,” *BIT* 24, 467–472.

G. E. Forsythe and G. H. Golub(1965). “On the Stationary Values of a Second-Degree Polynomial on the Unit Sphere,” *SIAM J. App. Math.* 14, 1050–1068.

L. Eldén(1980). "Perturbation Theory for the Least Squares Problem with Linear Equality Constraints," *SIAM J. Num, Anal.* 17, 338–350.

W. Gander(1981). "Least Squares with a Quadratic Constraint," *Numer. Math.* 36, 291–307.

L. Eldén(1983). "A. Wegihted Pseudoinverse, Generalized Singular Values, and Constrained Least Squares Problems," *BIT* 22, 487–502.

G. W. Stewart(1984). "On the Asymptotic Behavior of Scaled Singular Value and QR Decompositions," *Math. Comp.* 43, 483–490.

G. H. Golub and U. von Matt(1991). "Quadratically Constrained Least Squares and Quadratic Problems," *Numer. Math.* 59, 561–580.

T. F. Chan, J. A. Olkin, and D. Cooley(1992). "Solving Quadratically Constrained Least Squares Using Black Box Solvers," *BIT* 32, 481–495.

LSQI 问题涉及修正和处理带状稀疏矩阵的其他计算方面可见:

K. Schittkowski and J. Stoer(1979). "A Factorization Method for the Solution of Constrained Linear Least Squares Problems Allowing for Subsequent Data changes," *Numer. Math.* 31, 431–463.

D. P. O' Leary and J. A. Simmons(1981). "A Bidiagonalization-Regularization Procedure for Large Scale Discretizations of Ill-Posed Problems," *SIAM J. Sci, and Stat. Comp.* 2, 474–489.

Å. Björck(1984). "A General Updating Algorithm for Constrained Linear Least Squares Problems," *SIAM J. Sci. and Stat. Comp.* 5, 394–402.

L. Eldén(1984). "An Algorithm for the Regularization of Ill-Conditioned, Banded Least Squares Problems," *SIAM J. Sci. and Stat. Comp.* 5, 237–254.

关于 LSE 问题各个方面的讨论和分析可见:

M. J. D. Powell and J. K. Reid(1968). "On Applying Householder's Method to Linear Least Squares Problems," *Proc. IFIP Congress*, pp. 122–126.

C. Van Loan(1985). "On the Method of Weighting for Equality Constrained Least Squares Problems," *SIAM J. Numer. Anal.* 22, 851–864.

J. L. Barlow, N. K. Nichols, and R. J. Plemmons(1988). "Iterative Methods for Equality Constrained Least Squares Problems," SIAM J. Sci. and Stat. Comp. 9, 892–906.

J. L. Barlow(1988). "Error Analysis and Implementation Aspects of Deferred Correction for Equality Constrained Least-Squares Problems," *SIAM J. Num. Anal.* 25, 1340–1358.

J. L. Barlow and S. L. Handy(1988). "The Direct Solution of Weighted and Equality Constrained Least-Squares Problems," *SIAM J. Sci. Stat. Comp.* 9, 704–716.

J. L. Barlow and U. B. Vemulapati(1992). "A Note on Deferred Correction of Equality Constrained Least Squares Problem," *SIAM J. Num. Anal.* 29, 249–256.

M. Wei(1992). "Perturbation Theory for the Rank-Deficient Equality Constrained Least Squares Problem," *SIAM J. Num. Anal.* 29, 1462–1481.

M. Wei(1992). "Algebraic Properties of the Rank-Deficient Equality-Constrained and Weighted Least Squares Problems," *Lin. Alg. and Its Applic.* 161, 27–44.

M. Gulliksson and P-Å. Wedin(1992). "Modifying the QR-Decomposition to Constrained and Weighted Linear Least Squares," *SIAM J. Matrix Anal. Appl.* 13, 1298–1313.

Å. Björck and C. C. Paige(1994). "Solution of Augmented Linear Systems Using Orthogonal Factorizations," *BIT* 34, 1–24.

M. Gulliksson(1994). "Iterative Refinement for Constrained and Weighted Linear Least Squares," *BIT* 34, 239–253.

M. Gulliksson(1995). "Backward Error Analysis for the Constrained and Weighted Linear Least Squares Problem When Using the Weighted QR Factorization," *SIAM J. Matrix. Anal. Appl.* 13, 675–687.

广义分解对广义最小二乘问题起重要作用：

C. C. Paige(1985). "The General Linear Model and the Generalized Singular Value Decomposition," *Lin. Alg. and Its Applic.* 70, 269–284.

C. C. Paige(1990). "Some Aspects of Generalized QR Factorization." in *Reliable Numerical Computations,* M. Cox and S. Hammarling(eds), Clarendon Press, Oxford.

E. Anderson, Z. Bai, and J. Dongarra(1992). "Generalized QR Factorization and Its Applications," *Lin. Alg. and Its Applic.* 162/163/164, 243–271.

## 12.2 利用 SVD 选取子列集

正如 5.5 节所介绍的那样, 秩亏的 LS 问题 $\min\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2$ 的求解可以用

$$\boldsymbol{x}_{\tilde{r}}=\sum_{i=1}^{\tilde{r}}\frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\boldsymbol{v}_i,\quad \tilde{r}\leqslant r$$

来逼近极小范数解

$$\boldsymbol{x}_{LS}=\sum_{i=1}^{r}\frac{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{b}}{\sigma_i}\boldsymbol{v}_i,\quad r=\mathrm{rank}(\boldsymbol{A}),$$

其中

$$\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}=\sum_{i=1}^{r}\sigma_i\boldsymbol{u}_i\boldsymbol{v}_i^{\mathrm{T}} \tag{12.2.1}$$

为 $\boldsymbol{A}$ 的奇异值分解 (SVD), $\tilde{r}$ 为秩 $r$ 的数值上的估计. 注意, $x_{\tilde{r}}$ 极小化 $\|\boldsymbol{A}_{\tilde{r}}\boldsymbol{x}-\boldsymbol{b}\|_2$, 其中

$$\boldsymbol{A}_{\tilde{r}}=\sum_{i=1}^{\tilde{r}}\sigma_i\boldsymbol{u}_i\boldsymbol{v}_i^{\mathrm{T}}$$

为离 $\boldsymbol{A}$ 最近的秩为 $\tilde{r}$ 的矩阵, 见定理 2.5.3.

在 LS 问题中用 $\boldsymbol{A}_{\tilde{r}}$ 代替 $\boldsymbol{A}$ 相当于忽略了小的奇异值, 当 $\boldsymbol{A}$ 是从噪声数据中导出时, 这样处理是很有道理的. 然而, 在另外的应用中, 秩亏意味着在构成基本模型的因素中有冗余部分. 在这种情况下, 建模者对诸如 $\boldsymbol{A}_{\tilde{r}}\boldsymbol{x}_{\tilde{r}}$ 这样含有所有的 $n$ 个因素的预报可能不感兴趣. 相反, 可能需要预报量 $\boldsymbol{A}\boldsymbol{y}$, 其中 $\boldsymbol{y}$ 至多含有 $\tilde{r}$ 个非零

元素. 非零元的位置决定了 $\boldsymbol{A}$ 的哪些列即模型中的哪些因素, 用来逼近观察向量 $\boldsymbol{b}$. 如何挑选这些列是*子集选取*问题, 也是本节的主题.

本节的内容建立在 2.6 节和第 5 章的基础之上.

### 12.2.1　列选主的 QR 方法

列选主的 QR 算法可以看作是挑选 $\boldsymbol{A}$ 的线性无关子列的一种方法, 然后用这些子列来预测 $\boldsymbol{b}$. 假设我们对矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 用算法 5.4.1 计算出一正交矩阵 $\boldsymbol{Q}$ 和排列矩阵 $\boldsymbol{\Pi}$, 使得 $\boldsymbol{R} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{\Pi}$ 为上三角形矩阵. 如果 $R(1:\tilde{r}, 1:\tilde{r})z = \tilde{b}(1:\tilde{r})$, 其中 $\tilde{\boldsymbol{b}} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{b}$, 令

$$\boldsymbol{y} = \boldsymbol{\Pi}\begin{bmatrix} \boldsymbol{z} \\ 0 \end{bmatrix},$$

则 $\boldsymbol{A}\boldsymbol{y}$ 是 $\boldsymbol{b}$ 的用到矩阵 $\boldsymbol{A}\boldsymbol{\Pi}$ 的前 $\tilde{r}$ 列的一个近似 LS 预测值.

### 12.2.2　利用 SVD

虽然列选主的 QR 算法是处理秩亏问题的一种相当可靠的方法, 但正如 5.5 节所讨论的那样, 有时候 SVD 更可取. 因此, 我们介绍一下 Golub, Klema, and Stewart(1976) 提出的基于 SVD 的子列集选取算法, 整个方法如下:

- *计算* SVD, $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$, *并利用它确定秩的估计* $\tilde{r}$.
- *计算置换矩阵* $\boldsymbol{P}$, *使得矩阵* $\boldsymbol{B}_1 \in \mathbb{R}^{m\times\tilde{r}}$ *的列充分无关, 其中* $\boldsymbol{A}\boldsymbol{P} = [\boldsymbol{B}_1 \quad \boldsymbol{B}_2]$.
- *用向量* $\boldsymbol{A}\boldsymbol{y}$ *预测* $\boldsymbol{b}$, *这里* $\boldsymbol{y} = \boldsymbol{P}\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{0} \end{bmatrix}, \boldsymbol{z} \in \mathbb{R}^{\tilde{r}}$ *极小化* $\|\boldsymbol{B}_1\boldsymbol{z} - \boldsymbol{b}\|_2$.

第二步是关键. 从

$$\min_{\boldsymbol{z}\in\mathbb{R}^{\tilde{r}}} \|\boldsymbol{B}_1\boldsymbol{z} - \boldsymbol{b}\|_2 = \|\boldsymbol{A}\boldsymbol{y} - \boldsymbol{b}\|_2 \geqslant \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$$

可知置换矩阵 $\boldsymbol{P}$ 应该选得使余量 $(\boldsymbol{I} - \boldsymbol{B}_1\boldsymbol{B}_1^{+})\boldsymbol{b}$ 尽可能小. 不幸的是, 这样的求解算法是不稳定的. 例如, 若

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1+\varepsilon & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix},$$

$\tilde{r} = 2, \boldsymbol{P} = \boldsymbol{I}$, 则有 $\min\|\boldsymbol{B}_1\boldsymbol{z} - \boldsymbol{b}\|_2 = 0$, 但 $\|\boldsymbol{B}_1^{+}\boldsymbol{b}\|_2 = 0(1/\varepsilon)$. 另一方面, 任何包含 $\boldsymbol{A}$ 的第 3 列的真子列集都是线性无关性极强的, 但得到的残量却都很大.

这个例子表明在所选取列的无关性与它们所得到的残量的范数之间有冲突. 处理这种冲突需要用到关于 $\boldsymbol{B}_1$ 的最小奇异值 $\sigma_{\tilde{r}}(\boldsymbol{B}_1)$ 界的范数的数学性质.

**定理 12.2.1**　*设* $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ *的* SVD *由* (12.2.1) *给出*, $\tilde{r} \leqslant \mathrm{rank}(\boldsymbol{A})$, *定义矩阵* $\boldsymbol{B}_1 \in \mathbb{R}^{m\times\tilde{r}}$:

$$\boldsymbol{A}\boldsymbol{P} = [\underset{\tilde{r}}{\boldsymbol{B}_1} \quad \underset{n-\tilde{r}}{\boldsymbol{B}_2}],$$

*其中* $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ *为一置换矩阵. 如果*

$$
\boldsymbol{P}^{\mathrm{T}}\boldsymbol{V} = \begin{bmatrix} \tilde{\boldsymbol{V}}_{11} & \tilde{\boldsymbol{V}}_{12} \\ \tilde{\boldsymbol{V}}_{21} & \tilde{\boldsymbol{V}}_{22} \end{bmatrix} \begin{matrix} \tilde{r} \\ n-\tilde{r} \end{matrix} \qquad (12.2.2)
$$
$$
\qquad\quad \tilde{r} \qquad n-\tilde{r}
$$

且 $\tilde{\boldsymbol{V}}_{11}$ 非奇, 则

$$
\frac{\sigma_{\tilde{r}}(\boldsymbol{A})}{\|\tilde{\boldsymbol{V}}_{11}^{-1}\|_2} \leqslant \sigma_{\tilde{r}}(\boldsymbol{B}_1) \leqslant \sigma_{\tilde{r}}(\boldsymbol{A}).
$$

**证明** 从 8.6.1 节中给出的奇异值的极小化极大特征可知上界估计成立.

为了得到下界, 划分奇异值的对角矩阵:

$$
\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_2 \end{bmatrix} \begin{matrix} \tilde{r} \\ m-\tilde{r} \end{matrix}
$$
$$
\quad \tilde{r} \quad n-\tilde{r}
$$

如果 $\boldsymbol{w} \in \mathbb{R}^{\tilde{r}}$ 为一单位向量且 $\|\boldsymbol{B}_1\boldsymbol{w}\|_2 = \sigma_{\tilde{r}}(\boldsymbol{B})_1$, 则

$$
\begin{aligned}
\sigma_{\tilde{r}}(\boldsymbol{B}_1)^2 = \|\boldsymbol{B}_1\boldsymbol{w}\|_2^2 &= \left\|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}\boldsymbol{P}\begin{bmatrix} \boldsymbol{w} \\ 0 \end{bmatrix}\right\|_2^2 \\
&= \|\boldsymbol{\Sigma}_1\tilde{\boldsymbol{V}}_{11}^{\mathrm{T}}\boldsymbol{w}\|_2^2 + \|\boldsymbol{\Sigma}_2\tilde{\boldsymbol{V}}_{12}^{\mathrm{T}}\boldsymbol{w}\|_2^2.
\end{aligned}
$$

由 $\|\boldsymbol{\Sigma}_1\tilde{\boldsymbol{V}}_{11}^{\mathrm{T}}\boldsymbol{w}\|_2^2 \geqslant \sigma_{\tilde{r}}(\boldsymbol{A})/\|\tilde{\boldsymbol{V}}_{11}^{-1}\|_2$ 知定理成立. □

这个结果表明, 为了得到一充分无关的子列集, 我们要选择置换矩阵 $\boldsymbol{P}$ 使得所得到的子矩阵 $\tilde{\boldsymbol{V}}_{11}$ 尽可能良态. 一个直观的做法是计算矩阵 $[\boldsymbol{V}_{11}^{\mathrm{T}} \quad \boldsymbol{V}_{21}^{\mathrm{T}}]$ 的列选主 QR 分解, 这里

$$
\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}_{11} & \boldsymbol{V}_{12} \\ \boldsymbol{V}_{21} & \boldsymbol{V}_{22} \end{bmatrix} \begin{matrix} \tilde{r} \\ n-\tilde{r} \end{matrix}
$$
$$
\quad \tilde{r} \quad n-\tilde{r}
$$

是 (12.2.1) 中矩阵 $\boldsymbol{V}$ 的一分块形式. 特别地, 当我们用列选主的 QR 算法 (算法 5.4.1) 来计算

$$
\boldsymbol{Q}^{\mathrm{T}}[\boldsymbol{V}_{11}^{\mathrm{T}} \quad \boldsymbol{V}_{21}^{\mathrm{T}}]\boldsymbol{P} = [\underset{\tilde{r}}{\boldsymbol{R}_{11}} \quad \underset{n-\tilde{r}}{\boldsymbol{R}_{12}}]
$$

时, 从 (12.2.2) 可知

$$
\begin{bmatrix} \tilde{\boldsymbol{V}}_{11} \\ \tilde{\boldsymbol{V}}_{21} \end{bmatrix} = \boldsymbol{P}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{V}_{11} \\ \boldsymbol{V}_{21} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{11}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}} \\ \boldsymbol{R}_{12}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}} \end{bmatrix},
$$

其中 $\boldsymbol{Q}$ 为正交矩阵, $\boldsymbol{P}$ 为置换矩阵, $\boldsymbol{R}_{11}$ 为上三角形矩阵. 注意, $\boldsymbol{R}_{11}$ 是非奇的且 $\|\tilde{\boldsymbol{V}}_{11}^{-1}\|_2 = \|\boldsymbol{R}_{11}^{-1}\|_2$. 直观上, 列选主有利于得到一个良态的 $\boldsymbol{R}_{11}$, 且整个过程倾向于产生一良态的 $\tilde{\boldsymbol{V}}_{11}$. 因此, 我们有下列算法.

**算法 12.2.1** 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 和 $\boldsymbol{b} \in \mathbb{R}^m$, 本算法计算出一置换矩阵 $\boldsymbol{P}$, 估计秩 $\tilde{r}$ 和一向量 $\boldsymbol{z} \in \mathbb{R}^{\tilde{r}}$, 使得矩阵 $\boldsymbol{B} = \boldsymbol{A}\boldsymbol{P}$ 的前 $\tilde{r}$ 列无关且 $\|B(:,1:\tilde{r})\boldsymbol{z} - \boldsymbol{b}\|_2$ 达到极小.

计算 SVD: $U^{\mathrm{T}}AV=\mathrm{diag}(\sigma_1,\cdots,\sigma_n)$ 并存储 $V$.
确定 $\tilde{r}, \tilde{r} \leqslant \mathrm{rank}(A)$
用列选主的 QR 分解计算 $Q^{\mathrm{T}}V(:,1:\tilde{r})^{\mathrm{T}}P=[R_1 \quad R_{12}]$ 并令

$AP=[B_1 \quad B_2], B_1 \in \mathbb{R}^{m\times\tilde{r}}, B_2 \in \mathbb{R}^{m\times(n-\tilde{r})}$

求 $z \in \mathbb{R}^{\tilde{r}}$ 使得 $\|b-B_1z\|_2=\min$

**例 12.2.1**

$$A=\begin{bmatrix} 3 & 4 & 1.0001 \\ 7 & 4 & -3.0002 \\ 2 & 5 & 2.9999 \\ -1 & 4 & 5.0003 \end{bmatrix}, \quad b=\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

从 $\sigma_3(A)\approx 0.0001$ 的意义上讲, $A$ 接近于秩 2. 在算法 12.2.1 中令 $\tilde{r}=2$, 得解 $x=[0 \quad 0.236\,0 \quad -0.008\,5]^{\mathrm{T}}$, 残差 $\|Ax-b\|_2=0.196\,6$, 置换矩阵为 $P=[e_3 \quad e_2 \quad e_1]$. 注意 $x_{\mathrm{LS}}=[828.105\,6 \quad -827.856\,9 \quad 828.053\,6]^{\mathrm{T}}$, 相应的残差为 $\|Ax_{\mathrm{LS}}-b\|_2=0.034\,3$.

### 12.2.3　列无关与残差的进一步讨论

我们回过来讨论列的无关性与残差范数之间的冲突问题. 特别地, 为了评价上述子列集选取的方法, 我们观察一下向量 $y$ 的残差 $r_y=b-Ay=b_1-B_1z=(I-B_1B_1^+)b$. 这里 $B_1=B(:,1:\tilde{r}), B=AP$. 为此, 比较 $r_y$ 与 $r_{x_{\tilde{r}}}=b-Ax_{\tilde{r}}$ 是合适的, 因为 $A$ 已看成秩 $\tilde{r}$ 矩阵, 而 $x_{\tilde{r}}$ 为最近的秩 $\tilde{r}$ LS 问题 $\min\|A_{\tilde{r}}x-b\|_2$ 之解.

**定理 12.2.2**　如果 $r_y$ 与 $r_{x_{\tilde{r}}}$ 如上定义, $\tilde{V}_{11}$ 是 $P^TV$ 的 $\tilde{r}\times\tilde{r}$ 顺序主子矩阵, 则

$$\|r_{x_{\tilde{r}}}-r_y\|_2 \leqslant \frac{\sigma_{\tilde{r}+1}(A)}{\sigma_{\tilde{r}}(A)}\|\tilde{V}_{11}^{-1}\|_2\|b\|_2.$$

**证明**　注意 $r_{x_{\tilde{r}}}=(I-V_1U_1^{\mathrm{T}})b$ 与 $r_y=(I-Q_1Q_1^{\mathrm{T}})b$, 其中

$$U=\underset{\quad\ \ \tilde{r}\qquad\ m-\tilde{r}}{[\,U_1 \qquad U_2\,]}$$

是 (12.2.1) 中矩阵 $U$ 的一种分块形式, $Q_1=B_1(B_1^{\mathrm{T}}B_1)^{-\frac{1}{2}}$. 应用定理 2.6.1, 有

$$\|r_{x_{\tilde{r}}}-r_y\|_2 \leqslant \|U_1U_1^{\mathrm{T}}-Q_1Q_1^{\mathrm{T}}\|_2\|b\|_2=\|U_2^{\mathrm{T}}Q_1\|_2\|b\|_2.$$

再由定理 12.2.1 知

$$\|U_2^{\mathrm{T}}Q_1\|_2 \leqslant \|U_2^{\mathrm{T}}B_1\|_2\|(B_1^{\mathrm{T}}B_1)^{-1/2}\|_2 \leqslant \sigma_{\tilde{r}+1}(A)\frac{1}{\sigma_{\tilde{r}}(B_1)}$$
$$\leqslant \frac{\sigma_{\tilde{r}+1}(A)}{\sigma_{\tilde{r}}(A)}\|\tilde{V}_{11}^{-1}\|_2.$$

□

由 $\|r_{x_{\tilde{r}}} - r_y\|_2 = \|B_1 y - \sum_{i=1}^{\tilde{r}}(u_i^{\mathrm{T}} b)u_i\|_2$, 我们可以看出定理 12.2.2 给出了 $B_1 y$ 预测向量 $b$ 的 "稳定" 分量 (即 $U_1^{\mathrm{T}} b$) 的精确程度. 任何逼近 $U_2^{\mathrm{T}} b$ 的办法都会给出范数很大的解. 此外, 定理也说明, 若 $\sigma_{\tilde{r}+1}(A) \ll \sigma_{\tilde{r}}(A)$, 则任何合理的无关子列集实质上都会产生差不多大小的残量. 另一方面, 如果奇异值之间没有明显的差距, 则确定 $\tilde{r}$ 的值变得比较困难, 因而整个子列集的选取更加复杂.

### 习　题

**12.2.1**　已知 $A \in \mathbb{R}^{m\times n}, \|u^{\mathrm{T}} A\|_2 = \sigma, u^{\mathrm{T}} u = 1$, 证明: 若 $u^{\mathrm{T}}(Ax - b) = 0$, 对某个 $x \in \mathbb{R}^n, b \in \mathbb{R}^m$, 则 $\|x\|_2 \geqslant |u^{\mathrm{T}} b|/\sigma$.

**12.2.2**　证明: 若 $B_1 \in \mathbb{R}^{m\times k}$ 由 $A \in \mathbb{R}^{m\times n}$ 的 $k$ 列组成, 则 $\sigma_k(B_1) \leqslant \sigma_k(A)$.

**12.2.3**　在方程 (12.2.2) 中, 我们知道矩阵

$$P^{\mathrm{T}} V = \begin{bmatrix} \tilde{V}_{11} & \tilde{V}_{12} \\ \tilde{V}_{21} & \tilde{V}_{22} \end{bmatrix} \begin{matrix} \tilde{r} \\ n-\tilde{r} \end{matrix}$$
$$\qquad\quad \tilde{r} \qquad n-\tilde{r}$$

是正交的. 因此, 从 CS 分解 (定理 2.6.3) 知 $\|\tilde{V}_{11}^{-1}\|_2 = \|\tilde{V}_{22}^{-1}\|_2$. 试说明如何通过对 $[\tilde{V}_{22}^{\mathrm{T}} \quad \tilde{V}_{12}^{\mathrm{T}}]$ 用列选主的 QR 算法来计算矩阵 $P$. (对 $\tilde{r} > n/2$, 此方法比书中讨论的方法更经济). 将此观察用于算法 12.2.1.

### 本节注释与参考文献

本节的材料选自:

G. H. Golub, V. Klema and G. W. Stewart(1976). "Rank Degeneracy and Least Squares Problems," Technical Report TR-456, Department of Computer Science, University of Maryland, College Park, MD.

基于 12.3 节中的整体最小二乘插值技术的一个子列选取方法由下文给出:

S. Van Huffel and J. Vandewalle(1987). "Subset Selection Using the Total Least Squares Approach in Collinearity Problems with Errors in the Variables," *Lin, Alg. and Its Applic.* 88/89, 695–714.

关于子列集选取的文献浩如烟海，我们向读者推荐:

H. Hotelling(1957). "The Relations of the Newer Multivariate Statistical Methods to Factor Analysis," *Brit. J. Stat. Psych.* 10, 69–79.

## 12.3　整体最小二乘

极小化问题 $\min\|D(Ax-b)\|_2$, 其中 $A \in \mathbb{R}^{m\times n}, D = \mathrm{diag}(d_1, \cdots, d_m)$ 非奇, 可重新整理为:

$$\min_{b+r\in \mathrm{ran}(A)} \|Dr\|_2, \quad r \in \mathbb{R}^m. \tag{12.3.1}$$

在此问题中, 隐含地假定只是观察向量 $b$ 有误差. 当 "数据" $A$ 也有误差时, 则更自然地考虑如下问题:

$$\min_{b+r\in \mathrm{ran}(A+E)} \|D[E \quad r]T\|_F, \quad E\in\mathbb{R}^{m\times m}, \quad r\in\mathbb{R}^m, \tag{12.3.2}$$

其中 $D=\mathrm{diag}(d_1,d_2,\cdots,d_m)$ 和 $T=\mathrm{diag}(t_1,t_2,\cdots,t_{n+1})$ 都非奇异. Golub and Van Loan(1980) 讨论了这类问题, 称其为**整体最小二乘** (TLS) 问题.

若能找到 (12.3.2) 的一个极小点 $[E_0 \quad r_0]$, 则任何满足 $(A+E_0)x=b+r_0$ 的 $x$ 都称为 TLS 解. 然而, 应该认识到 (12.3.2) 可能根本无解. 例如, 若

$$A=\begin{bmatrix}1&0\\0&0\\0&0\end{bmatrix},\quad b=\begin{bmatrix}1\\1\\1\end{bmatrix},\quad D=I_3,\quad T=I_3,\quad E_\varepsilon=\begin{bmatrix}0&0\\0&\varepsilon\\0&\varepsilon\end{bmatrix},$$

则对任何 $\varepsilon>0, b\in\mathrm{ran}(A+E_\varepsilon)$. 但是对 $b+r\in\mathrm{ran}(A+E)$, $\|[E \quad r]\|_F$ 无最小值.

我们可以把 (12.3.2) 推广到有多个右端项的情形. 特别地, 对 $B\in\mathbb{R}^{m\times k}$, 我们得到问题

$$\min_{\mathrm{ran}(B+R)\subseteq\mathrm{ran}(A+E)} \|D[E \quad R]T\|_F, \tag{12.3.3}$$

其中, $E\in\mathbb{R}^{m\times n}, R\in\mathbb{R}^{m\times k}$, 且矩阵 $D=\mathrm{diag}(d_1,\cdots,d_m)$ 和 $T=\mathrm{diag}(t_1,\cdots,t_{n+k})$ 非奇异. 若 $[E_0 \quad R_0]$ 是问题 (12.3.3) 的解, 则任何满足 $(A+E_0)X=(B+R_0)$ 的解 $X\in\mathbb{R}^{n\times k}$ 都称为 (12.3.3) 的 TLS 解.

本节讨论 TLS 问题的一些数学性质, 并证明可用 SVD 来求解. 它需要第 5 章作预备知识. Van Huffel and Vanderwalle(1991) 的专著对 TLS 作了详细的讨论.

### 12.3.1 数学基础

下面的定理给出了多右端向量的 TLS 解的存在性和唯一条件:

**定理 12.3.1** 设 $A,B,D$ 和 $T$ 如上所述, 且假定 $m\geqslant n+k$, 作划分

$$C=D[A \quad B]T=[\underset{n}{C_1} \quad \underset{k}{C_2}],$$

它的 SVD 形式为: $U^{\mathrm{T}}CV=\mathrm{diag}(\sigma_1,\cdots,\sigma_{n+k})=\Sigma$, 其中

$$U=[\underset{n}{U_1} \quad \underset{k}{U_2}],\quad V=\begin{bmatrix}V_{11}&V_{12}\\V_{21}&V_{22}\end{bmatrix}\begin{matrix}n\\k\end{matrix},\quad \Sigma=\begin{bmatrix}\Sigma_1&0\\0&\Sigma_2\end{bmatrix}\begin{matrix}n\\k.\end{matrix}$$
$$\qquad\qquad\qquad\qquad\quad n \qquad k \qquad\qquad\qquad\quad n \qquad k$$

如果 $\sigma_n(C_1)>\sigma_{n+1}(C)$, 则由

$$D[E_0 \quad R_0]T=-U_2\Sigma_2[V_{12}^{\mathrm{T}} \quad V_{22}^{\mathrm{T}}] \tag{12.3.4}$$

定义的矩阵 $[E_0,R_0]$ 为 (12.3.3) 的解. 若 $T_1=\mathrm{diag}(t_1,\cdots,t_n), T_2=\mathrm{diag}(t_{n+1},\cdots,t_{n+k})$, 则矩阵

$$X_{\rm TLS} = -T_1 V_{12} V_{22}^{-1} T_2^{-1}$$

存在且是方程 $(A + E_0)X = B + R_0$ 的唯一解.

**证明** 我们先从假设 $\sigma_n(C_1) > \sigma_{n+1}(C)$ 得出两个结果. 由方程 $CV = U\Sigma$ 知 $C_1V_{12} + C_2V_{22} = U_2\Sigma_2$. 我们希望证明 $V_{22}$ 是非奇的. 假设存在单位 2 范数 $x$ 使得 $V_{22}x = 0$. 从 $V_{12}^{\rm T}V_{12} + V_{22}^{\rm T}V_{22} = I$ 知 $\|V_{12}x\|_2 = 1$. 但是 $\sigma_{n+1}(C) \geqslant \|U_2\Sigma_2 x\|_2 = \|C_1V_{12}x\|_2 \geqslant \sigma_n(C_1)$, 这与假设矛盾. 因此子矩阵 $V_{22}$ 非奇异.

从 $\sigma_n(C_1) > \sigma_{n+1}(C)$ 可得出的另外一个事实是关于 $\sigma_n(C)$ 与 $\sigma_{n+1}(C)$ 的严格分离. 由推论 8.3.3 知 $\sigma_n(C) \geqslant \sigma_n(C_1)$, 因而 $\sigma_n(C) \geqslant \sigma_n(C_1) > \sigma_{n+1}(C)$.

现在我们来证明定理. 如果 $\mathrm{ran}(B+R) \subset \mathrm{ran}(A+E)$, 则存在 $X \in \mathbb{R}^{n\times k}$ 使得 $(A+E)X = B+R$, 即

$$\{D[A \quad B]T + D[E \quad R]T\}T^{-1}\begin{bmatrix} X \\ -I_k \end{bmatrix} = 0. \tag{12.3.5}$$

因此在大括号内的矩阵之秩不超过 $n$. 从定理 2.5.3 的讨论中可证

$$\|D[E \quad R]T\|_F \geqslant \sum_{i=n+1}^{n+k} \sigma_i(C)^2,$$

且当 $[E \quad R] = [E_0 \quad R_0]$ 时可达到下界. 不等式 $\sigma_n(C) > \sigma_{n+1}(C)$ 保证了 $[E_0 \quad R_0]$ 是唯一的极小点. 矩阵

$$\{D[A \quad B]T + D[E_0 \quad R_0]T\} = U_1\Sigma_1[V_{11}^{\rm T} \quad V_{21}^{\rm T}]$$

的零空间是 $\begin{bmatrix} V_{12} \\ V_{22} \end{bmatrix}$ 的象空间. 因此, 从 (12.3.5) 可知, 对某个 $k\times k$ 矩阵 $S$, 有

$$T^{-1}\begin{bmatrix} X \\ -I_k \end{bmatrix} = \begin{bmatrix} V_{12} \\ V_{22} \end{bmatrix} S.$$

从方程 $T_1^{-1}X = V_{12}S$ 和 $-T_2^{-1} = V_{22}S$ 可看出 $S = -V_{22}^{-1}T_2^{-1}$. 因此必有

$$X = T_1V_{12}S = -T_1V_{12}V_{22}^{-1}T_2^{-1} = X_{\rm TLS}.$$

□

如果 $\sigma_n(C) = \sigma_{n+1}(C)$, 则 TLS 问题仍可能有解, 虽然不一定是唯一的. 在这种情况下, 希望找到它们当中的 "极小范数" 解. 为此, 在空间 $\mathbb{R}^{n\times k}$ 上定义 $\tau$ 范数: $\|Z\|_\tau = \|T_1^{-1}ZT_2\|_2$. 如果 $X$ 由 (12.3.5) 给出, 则从 CS 分解 (定理 2.6.3) 可知

$$\|X\|_\tau^2 = \|V_{12}V_{22}^{-1}\|_2^2 = (1 - \sigma_k(V_{22})^2)/\sigma_k(V_{22})^2.$$

这表明在定理 12.3.1 中应选取 $V$ 使得 $\sigma_k(V_{22})$ 达到极大.

### 12.3.2 $k = 1$ 时的计算

现介绍一下当 $k=1$ 这一重要情形时如何使 $V_{22}$ 极大. 假定矩阵 $C$ 的奇异值满足 $\sigma_{n-p} > \sigma_{n-p+1} = \cdots = \sigma_{n+1}$. 对 $V$ 作列划分: $V = [v_1 \quad v_2 \quad \cdots, \quad v_{n+1}]$. 若 $\tilde{Q}$ 为 Householder 矩阵, 使得

$$V(:,n+1-p:n+1)\tilde{\boldsymbol{Q}} = \begin{bmatrix} \boldsymbol{W} & \boldsymbol{z} \\ \boldsymbol{0} & \alpha \end{bmatrix} \begin{matrix} n \\ 1 \end{matrix}$$
$$\qquad\qquad\qquad\qquad p \quad 1$$

则在空间 $\mathrm{span}\{\boldsymbol{v}_{n+1-p},\cdots,\boldsymbol{v}_{n+1}\}$ 的所有向量中, $\begin{bmatrix}\boldsymbol{z}\\ \alpha\end{bmatrix}$ 的第 $n+1$ 个分量最大. 若 $\alpha=0$, 则 TLS 问题无解. 否则, $x_{\mathrm{TLS}} = -\boldsymbol{T}_1\boldsymbol{z}/(t_{n+1}\alpha)$. 此外,

$$\begin{bmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q} \end{bmatrix} \boldsymbol{U}^{\mathrm{T}}(\boldsymbol{D}[\boldsymbol{A} \quad \boldsymbol{b}]\boldsymbol{T})\boldsymbol{V} \begin{bmatrix} \boldsymbol{I}_{n-p} & \boldsymbol{0} \\ \boldsymbol{0} & \tilde{\boldsymbol{Q}} \end{bmatrix} = \boldsymbol{\Sigma},$$

$$\boldsymbol{D}[\boldsymbol{E}_0 \quad \boldsymbol{r}_0]\boldsymbol{T} = -\boldsymbol{D}[\boldsymbol{A} \quad \boldsymbol{b}]\boldsymbol{T}\begin{bmatrix}\boldsymbol{z}\\ \alpha\end{bmatrix}[\boldsymbol{z}^{\mathrm{T}} \quad \alpha].$$

综上所述, 有下面的算法.

**算法 12.3.1** 给定矩阵 $\boldsymbol{A}\in\mathbb{R}^{m\times n}(m>n), b\in\mathbb{R}^m$ 及非奇异矩阵 $\boldsymbol{D}=\mathrm{diag}(d_1,d_2,\cdots,d_m)$, $\boldsymbol{T}=\mathrm{diag}(t_1,t_2,\cdots,t_{n+1})$. 本算法计算出 (如果有) 一向量 $x_{\mathrm{TLS}}\in\mathbb{R}^n$ 使得 $(\boldsymbol{A}+\boldsymbol{E}_0)\boldsymbol{x}=\boldsymbol{b}+\boldsymbol{r}_0$ 且 $\|\boldsymbol{D}[\boldsymbol{E}_0 \quad \boldsymbol{r}_0]\boldsymbol{T}\|_F$ 极小.

计算 SVD: $\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{D}[\boldsymbol{A} \quad b]\boldsymbol{T})\boldsymbol{V}=\mathrm{diag}(\sigma_1,\cdots,\sigma_{n+1})$ 并存储 $V$.

找出 $p$ 满足 $\sigma_1\geqslant\cdots\geqslant\sigma_{n-p}>\sigma_{n-p+1}=\cdots=\sigma_{n+1}$.

找出 Householder 矩阵 $\boldsymbol{P}$, 使得若 $\tilde{\boldsymbol{V}}=\boldsymbol{V}\boldsymbol{P}$, 则

  $\tilde{V}(n+1,n-p+1:n)=0$

**if** $\tilde{v}_{n+1,n+1}\neq 0$

  **for** $i=1:n$

    $x_i=-t_i\tilde{v}_{i,n+1}/(t_{n+1}\tilde{v}_{n+1,n+1})$

  **end**

**end**

此算法需要大约 $2mn^2+12n^3$ 个 flop, 主要花费在 SVD 的计算上.

**例 12.3.1** TLS 问题 $\min\limits_{(a+e)x=b+r}\|[e,r]\|_F$, 其中 $\boldsymbol{a}=[1,2,3,4]^{\mathrm{T}}, \boldsymbol{b}=[2.01,3.99,5.80,8.30]^{\mathrm{T}}$, 解为 $x_{\mathrm{TLS}}=2.0212$, $\boldsymbol{e}=[-0.0045,-0.0209,-0.1048,0.0855]^{\mathrm{T}}$, $\boldsymbol{r}=[0.0022,0.0103,0.0519,-0.0423]^{\mathrm{T}}$. 注意, 对这组数据, LS 解为 $x_{\mathrm{LS}}=2.0197$.

### 12.3.3 几何解释

可以证明, TLS 问题的解 $x_{\mathrm{TLS}}$ 使得

$$\psi(x)=\sum_{i=1}^{m} d_i^2\frac{|\boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{x}-b_i|^2}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{T}_1^{-2}\boldsymbol{x}+t_{n+1}^{-2}}$$

达到极小, 这里 $\boldsymbol{a}_i^{\mathrm{T}}$ 为 $\boldsymbol{A}$ 的第 $i$ 行, $b_i$ 是 $\boldsymbol{b}$ 的第 $i$ 个分量. 从这里可看出 TLS 问题的几何意义. 事实上, 数

$$\frac{|\boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{x}-b_i|^2}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{T}_1^{-2}\boldsymbol{x}+t_{n+1}^{-2}}$$

是向量 $\begin{bmatrix} a_i \\ b_i \end{bmatrix} \in \mathbb{R}^{n+1}$ 到子空间

$$\boldsymbol{P}_x = \left\{ \begin{bmatrix} \boldsymbol{a} \\ b \end{bmatrix} : \boldsymbol{a} \in \mathbb{R}^n, b \in \mathbb{R}, b = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{a} \right\}$$

中最近的点的距离之平方. 用到的范数是 $\|\boldsymbol{Z}\| = \|\boldsymbol{T}\boldsymbol{z}\|_2$. 这方面有大量的文章, 见 Pearson(1901) 和 Madansky(1959).

## 习　题

**12.3.1**　考虑 TLS 问题 (12.3.2), 矩阵 $\boldsymbol{D}$ 和 $\boldsymbol{T}$ 非奇异. (a) 证明：若 rank$(\boldsymbol{A}) < n$, 则 (12.3.2) 有解当且仅当 $\boldsymbol{b}$ ∈ran$(\boldsymbol{A})$. (b) 证明：若 rank$(\boldsymbol{A}) = n$, 则在 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{b} = 0$ 且 $|t_{n+1}|\|\boldsymbol{D}\boldsymbol{b}\|_2 \geqslant \sigma_n(\boldsymbol{D}\boldsymbol{A}\boldsymbol{T}_1)$ 时 (12.3.2) 无解, 这里 $\boldsymbol{T}_1$=diag$(t_1, \cdots, t_n)$.

**12.3.2**　证明：若 $\boldsymbol{C} = \boldsymbol{D}[\boldsymbol{A} \quad \boldsymbol{b}]\boldsymbol{T} = [\boldsymbol{A}_1 \quad \boldsymbol{d}]$ 且 $\sigma_n(\boldsymbol{C}) > \sigma_{n+1}(\boldsymbol{C})$, 则 TLS 解 $x$ 满足 $(\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1 - \sigma_{n+1}(\boldsymbol{C})^2\boldsymbol{I})\boldsymbol{x} = \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{d}$.

**12.3.3**　说明在极小值矩阵 $\boldsymbol{E}$ 的前 $p$ 列为零的附加约束下如何解 (12.3.2).

## 本节注释与参考文献

本节内容建立在下列文献的基础上：

G. H. Golub and C. F. Van Loan (1980). "An Analysis of the Total Least Squares Problem," *SIAM J. Num. Anal.* 17, 883–893.

基于 SVD 的 TLS 问题由下文提出：

G. H. Golub and C. Reinsch (1970). "Singular Value Decomposition and Least Squares Solutions," *Numer. Math.* 14, 403–420.

G. H. Golub (1973). "Some Modified Matrix Eigenvalue Problems," *SIAM Review* 15, 318–334.

关于 TLS 问题最详细的研究见：

S. Van Huffel and J. Vandewalle (1991). *The Total Least Squares Problem: Computational Aspects and Analysis,* SIAM Publications, Philadelphia.

若已知 $\boldsymbol{A}$ 的某些列是精确的, 则在 TLS 扰动矩阵 $\boldsymbol{E}$ 中要求这些列为 0 是合理的. 关于这种约束的 TLS 问题的讨论可见：

J. W. Demmel (1987). "The Smallest Perturbation of a Submatrix which Lowers the Rank and Constrained Total Least Squares Problems," *SIAM J. Numer. Anal.* 24, 199–206.

S. Van Huffel and J. Vandewalle (1988). "The Partial Total Least Squares Algorithm," *J. Comp. and App. Math.* 21, 333–342.

S. Van Huffel and J. Vandewalle (1988). "Analysis and Solution of the Nongeneric Total Least Squares Problem," *SIAM J. Matrix Anal. Appl.* 9, 360–372.

S. Van Huffel and J. Vandewalle (1989). "Analysis and Properties of the Generalized Total Least Squares Problem $AX \approx B$ When Some or All Columns in $A$ are Subject to Error," *SIAM J. Matrix Anal. Appl.* 10, 294–315.

S. Van Huffel and H. Zha (1991). "The Restricted Total Least Squares Problem: Formulation. Algorithm, and Properties," *SIAM J. Matrix Anal. Appl.* 12, 292–309.

S. Van Huffel (1992). "On the Significance of Nongeneric Total Least Squares Problems," *SIAM J. Matrix Anal. Appl.* 13, 20–35.

M. Wei (1992). "The Analysis for the Total Least Squares Problem with More than One Solution," *SIAM J. Matrix Anal. Appl.* 13, 746–763.

S. Van Huffel and H. Zha (1993). "An Efficient Total Least Squares Algorithm Based On a Rank-Revealing Two-Sided Orthogonal Decomposition," *Numerical Algorithms* 4, 101–133.

C. C. Paige and M. Wei (1993). "Analysis of the Generalized Total Least Squares Problem $AX = B$ when Some of the Columns are Free of Error," *Numer. Math.* 65, 177–202.

R. D. Fierro and J. R. Bunch (1994). "Collinearity and Total Least Squares," *SIAM J. Matrix Anal. Appl.* 15, 1167–1181.

当数据矩阵有误差时，涉及最小二乘插值的其他文献有：

K. Pearson (1901). "On Lines and Planes of Closest Fit to Points in Space," *Phil. Mag.* 2, 559–572.

A. Wald (1940). "The Fitting of Straight Lines if Both Variables are Subject to Error," *Annals of Mathematical Statistics* 11, 284–300.

A. Madansky (1959). "The Fitting of Straight Lines When Both Variables Are Subject to Error," *J. Amer. Stat. Assoc.* 54, 173–205.

I. Linnik (1961). *Method of Least Squares and Principles of the Theory of Observations*, Pergamon Press, New York.

W. G. Cochrane (1968). "Errors of Measurement in Statistics," *Technometrics* 10, 637–666.

R. F. Gunst, J. T. Webster, and R. L. Mason (1976). "A Comparison of Least Squares and Latent Root Regression Estimators," *Technometrics* 18. 75–83.

G. W. Stewart (1977c). "Sensitivity Coefficients for the Effects of Errors in the Independent Variables in a Linear Regression," Technical Report TR-571, Department of Computer Science, University of Maryland, College Park, MD.

A. Van der Sluis and G. W. Veltkamp (1979). "Restoring Rank and Consistency by Orthogonal Projection," *Lin. Alg. and Its Applic.* 28, 257–278.

## 12.4 利用 SVD 计算子空间

有时候需要了解两个给定的子空间之间的关系. 他们有多近? 它们有交吗? 它们中的一个可旋转成另一个吗? 等等. 本节将阐明如何用 SVD 来回答诸如此类的问题. 这里需要第 5 章和 8.6 节的知识.

### 12.4.1 子空间的旋转

假设 $\boldsymbol{A} \in \mathbb{R}^{m\times p}$ 是经过一系列实验所获得的数据矩阵. 如果这些实验再重复

一遍, 则得到另一数据矩阵 $\boldsymbol{B}\in\mathbb{R}^{m\times p}$. 在正交 Procrustes 问题中, 要知道 $B$ 能否旋转成 $\boldsymbol{A}$, 可求解如下问题:

$$\min\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{Q}\|_F,\ \text{在约束}\ \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}=\boldsymbol{I}_p\ \text{下}. \tag{12.4.1}$$

我们知道, 矩阵的迹是它对角线元素之和, 因此有 $\operatorname{tr}(\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C})=\|\boldsymbol{C}\|_F^2$. 由 $\boldsymbol{Q}$ 的正交性有

$$\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{Q}\|_F^2=\operatorname{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})+\operatorname{tr}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B})-2\operatorname{tr}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}).$$

因此, (12.4.1) 等价于使 $\operatorname{tr}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A})$ 极大化.

使 $\operatorname{tr}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A})$ 极大的 $\boldsymbol{Q}$ 可通过计算 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}$ 的 SVD 求出. 若 $\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A})\boldsymbol{V}=\boldsymbol{\Sigma}=\operatorname{diag}(\sigma_1,\cdots,\sigma_p)$ 为矩阵的 SVD, 定义正交矩阵 $Z=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{U}$, 则

$$\operatorname{tr}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A})=\operatorname{tr}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}})=\operatorname{tr}(\boldsymbol{Z}\boldsymbol{\Sigma})=\sum_{i=1}^{p}z_{ii}\sigma_i\leqslant\sum_{i=1}^{p}\sigma_i.$$

显然, 当取 $\boldsymbol{Q}=\boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}$ 时, $\boldsymbol{Z}=\boldsymbol{I}_p$, 上界达到. 由此可得下述算法.

**算法 12.4.1** 已知 $\boldsymbol{A}$ 和 $\boldsymbol{B}\in\mathbb{R}^{m\times p}$, 本算法找出使得 $\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{Q}\|_F$ 达到极小的正交矩阵 $\boldsymbol{Q}\in\mathbb{R}^{p\times p}$

$\boldsymbol{C}=\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}$

计算 SVD: $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{V}=\boldsymbol{\Sigma}$, 存储 $\boldsymbol{U}$ 和 $\boldsymbol{V}$

$\boldsymbol{Q}=\boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}$

解矩阵 $\boldsymbol{Q}$ 是 $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}$ 的正交极因子, 见 4.2.10 节.

**例 12.4.1**

$$\boldsymbol{Q}=\begin{bmatrix}0.9999 & -0.0126\\ 0.0126 & 0.9999\end{bmatrix}$$

使

$$\left\|\begin{bmatrix}1&2\\3&4\\5&6\\7&8\end{bmatrix}\boldsymbol{Q}-\begin{bmatrix}1.2&2.1\\2.9&4.3\\5.2&6.1\\6.8&8.1\end{bmatrix}\right\|_F$$

达到极小.

### 12.4.2 零空间的交集

设 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 和 $\boldsymbol{B}\in\mathbb{R}^{p\times n}$ 给定, 考虑如何寻找 $\operatorname{null}(\boldsymbol{A})\cap\operatorname{null}(\boldsymbol{B})$ 的一组标准正交基. 一种方法是计算矩阵

$$\boldsymbol{C}=\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{B}\end{bmatrix}$$

的零空间, 因为 $\boldsymbol{C}x=0\Leftrightarrow x\in\operatorname{null}(\boldsymbol{A})\cap\operatorname{null}(\boldsymbol{B})$. 然而, 利用下面的定理可得到一个更经济的算法.

**定理 12.4.1** 设 $A\in\mathbb{R}^{m\times n}$, $\{z_1,\cdots,z_t\}$ 为 null($A$) 的一组标准正交基. 定义 $Z=\{z_1,\cdots,z_t\}$, 令 $\{w_1,\cdots,w_q\}$ 为 null($BZ$) 的一组标准正交基, 其中 $B\in\mathbb{R}^{p\times n}$. 若 $W=[w_1,\cdots,w_q]$, 则 $ZW$ 的列构成了 null($A$)∩null($B$) 的一组标准正交基.

**证明** 因为 $AZ=0$ 且 $(BZ)W=0$, 故 ran($ZW$) ⊂null($A$)∩null($B$). 设 $x\in$null($A$)∩null($B$), 则存在不为 0 的 $a\in\mathbb{R}^t$, 使得 $x=Za$. 但是, 由 $0=Bx=BZa$ 知存在 $b\in\mathbb{R}^q$ 使得 $a=Wb$. 所以 $x=ZWb\in$ran($ZW$). □

**算法 12.4.2** 给定 $A\in\mathbb{R}^{m\times n}$, $B\in\mathbb{R}^{p\times n}$, 本算法计算出一整数 $s$ 及矩阵 $Y=[y_1,\cdots,y_s]$, 使得 $Y$ 的列正交且张成空间 null($A$)∩null($B$). 若交是平凡空间, 则 $s=0$.

```
计算 SVD: U_A^T A V_A = diag(σ_i), 存储 V_A, 并令 r = rank(A).
if r < n
    C = B V_A(:, r+1 : n)
    计算 SVD: U_C^T C V_C = diag(γ_i), 存储 V_C, 并令 q = rank(C)
    if q < n − r
        s = n − r − q
        Y = V_A(:, r+1 : n) V_C(: q+1 : n−r)
    else
        s = 0
    end
else
    s = 0
end
```

此算法的计算工作量依赖于数 $m,n,p$ 和 $r$ 的相对大小.

我们指出, 该算法的实现需要一种决定何时忽略计算的奇异值 $\hat{\sigma}_i$ 的策略. 当用容许值 $\delta$(例如, $\hat{\sigma}_i<\delta\Rightarrow\hat{\sigma}_i=0$) 时, 意味着在 $\|A\hat{Y}\|_2\approx\|B\hat{Y}\|_2\approx\delta$ 的意义下计算值 $\hat{Y}$ 的列 "几乎" 定义了 $A$ 和 $B$ 的共同零空间.

**例 12.4.2** 若

$$A=\begin{bmatrix}1&-1&1\\1&-1&1\\1&-1&1\end{bmatrix}\quad 和\quad B=\begin{bmatrix}4&2&0\\2&1&0\\6&3&0\end{bmatrix},$$

则 null($A$)∩null($B$)=span$\{x\}$, $x=[1\quad -2\quad -3]^{\mathrm{T}}$. 用算法 12.4.2 计算, 得

$$V_{2A}V_{2C}=\begin{bmatrix}-0.8165&0.0000\\-0.4082&0.7071\\-0.4082&0.7071\end{bmatrix}\begin{bmatrix}-0.3273\\-0.9449\end{bmatrix}\approx\begin{bmatrix}0.2673\\-0.5345\\-0.8018\end{bmatrix}\approx 0.2673\begin{bmatrix}1\\-2\\-3\end{bmatrix}.$$

### 12.4.3 子空间之间的夹角

设 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 为 $\mathbb{R}^m$ 中的子空间, 它们的维数满足

$$p = \dim(\boldsymbol{F}) \geqslant \dim(\boldsymbol{G}) = q \geqslant 1.$$

$\boldsymbol{F}$ 和 $\boldsymbol{G}$ 之间的*主角* $\theta_1, \cdots, \theta_q \in [0, \pi/2]$ 定义如下:

$$\cos(\theta_k) = \max_{\boldsymbol{u}\in F} \max_{\boldsymbol{v}\in G} \boldsymbol{u}^{\mathrm{T}}\boldsymbol{v} = \boldsymbol{u}_k^{\mathrm{T}}\boldsymbol{v}_k,$$

约束条件为:

$$\begin{aligned} &\|\boldsymbol{u}\| = \|\boldsymbol{v}\| = 1 \\ &\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}_i = 0 \qquad i = 1 : k-1 \\ &\boldsymbol{v}^{\mathrm{T}}\boldsymbol{v}_i = 0 \qquad i = 1 : k-1. \end{aligned}$$

注意, 主角满足 $0 \leqslant \theta_1 \leqslant \cdots \leqslant \theta_q \leqslant \pi/2$. 向量 $\{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_q\}$ 和 $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_q\}$ 称为空间 $F$ 与 $G$ 之间的*主向量*.

主角和主向量问题在很多重要的统计应用中都会涉及到. 最大的主角与我们在 2.6.3 节中讨论过的等维数子空间之间的距离有关系. 若 $p = q$, 则 $\mathrm{dist}(\boldsymbol{F}, \boldsymbol{G}) = \sqrt{1 - \cos^2(\theta_p)} = \sin(\theta_p)$.

如果 $\boldsymbol{Q}_F \in \mathbb{R}^{m\times p}$ 和 $\boldsymbol{Q}_G \in \mathbb{R}^{m\times q}$ 的列分别是 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 的标准正交基, 则

$$\max_{\substack{\boldsymbol{u}\in F \\ \|\boldsymbol{u}\|_2=1}} \max_{\substack{\boldsymbol{v}\in G \\ \|\boldsymbol{v}\|_2=1}} \boldsymbol{u}^{\mathrm{T}}\boldsymbol{v} = \max_{\substack{\boldsymbol{y}\in\mathbb{R}^p \\ \|\boldsymbol{y}\|_2=1}} \max_{\substack{\boldsymbol{z}\in\mathbb{R}^q \\ \|\boldsymbol{z}\|_2=1}} \boldsymbol{y}^{\mathrm{T}}(\boldsymbol{Q}_F^{\mathrm{T}}\boldsymbol{Q}_G)\boldsymbol{z}.$$

从定理 8.6.1 给出的奇异值的极小化极大特征中可知, 若 $\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{Q}_F^{\mathrm{T}}\boldsymbol{Q}_G)\boldsymbol{Z} = \mathrm{diag}(\sigma_1, \cdots, \sigma_q)$ 是 $\boldsymbol{Q}_F^{\mathrm{T}}\boldsymbol{Q}_G$ 的 SVD, 则可定义 $\boldsymbol{u}_k, \boldsymbol{v}_k, \theta_k$ 如下:

$$\begin{aligned} [\boldsymbol{u}_1, \cdots \boldsymbol{u}_p] &= \boldsymbol{Q}_F\boldsymbol{Y}, \\ [\boldsymbol{v}_1, \cdots \boldsymbol{v}_q] &= \boldsymbol{Q}_G\boldsymbol{Z}, \\ \cos(\theta_k) &= \sigma_k, \quad k = 1 : q. \end{aligned}$$

一般地, 空间 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 都是某给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times p}$ 和 $\boldsymbol{B} \in \mathbb{R}^{m\times q}$ 的象空间. 在这种情况下, 所要的标准正交基可以通过计算这两个矩阵的 QR 分解得到.

**算法 12.4.3** 给定矩阵 $\boldsymbol{A} \in \mathbb{R}^{m\times p}$ 和 $\boldsymbol{B} \in \mathbb{R}^{m\times q} (p \geqslant q)$, 每个都是列线性无关的, 本算法计算出正交矩阵 $\boldsymbol{U} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_q], \boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_q]$ 及 $\cos(\theta_1), \cdots, \cos(\theta_q)$, 其中 $\theta_k$ 为 $\mathrm{ran}(\boldsymbol{A})$ 与 $\mathrm{ran}(\boldsymbol{B})$ 之间的主角, $\boldsymbol{u}_k$ 和 $\boldsymbol{v}_k$ 为相应的主向量.

用算法 5.2.1 计算 QR 分解

$$\begin{aligned} &\boldsymbol{A} = \boldsymbol{Q}_A\boldsymbol{R}_A, \quad \boldsymbol{Q}_A^{\mathrm{T}}\boldsymbol{Q}_A = \boldsymbol{I}_p, \ \boldsymbol{R}_A \in \mathbb{R}^{p\times p} \\ &\boldsymbol{B} = \boldsymbol{Q}_B\boldsymbol{R}_B, \quad \boldsymbol{Q}_B^{\mathrm{T}}\boldsymbol{Q}_B = \boldsymbol{I}_q, \ \boldsymbol{R}_B \in \mathbb{R}^{q\times q} \\ &\boldsymbol{C} = \boldsymbol{Q}_A^{\mathrm{T}}\boldsymbol{Q}_B. \end{aligned}$$

计算奇异值分解: $\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{Z} = \mathrm{diag}(\cos(\theta_k))$

$$Q_A Y(:,1:q) = [u_1, \cdots, u_q]$$
$$Q_B Z = [v_1, \cdots, v_q].$$

本算法约需 $4m(q^2+2p^2)+2pq(m+q)+12q^3$ 个 flop.

利用 SVD 计算主角和主向量的思想出现在 Björk and Golub(1973) 中. 该论文还讨论了 $A$ 和 $B$ 秩亏的情况.

### 12.4.4 子空间的交

算法 12.4.3 也可用来计算 $\mathrm{ran}(A)\cap\mathrm{ran}(B)$ 的标准正交基, 其中 $A\in\mathbb{R}^{m\times p}$, $B\in\mathbb{R}^{m\times q}$.

**定理 12.4.2** 设 $\{\cos(\theta_k), u_k, v_k\}_{k=1}^{q}$ 为由算法 12.4.3 产生的主角和主向量. 如果指标 $s$ 定义为 $1=\cos(\theta_1)=\cdots=\cos(\theta_s)>\cos(\theta_{s+1})$, 则有

$$\mathrm{ran}(A)\cap\mathrm{ran}(B)=\mathrm{span}\{u_1,\cdots,u_s\}=\mathrm{span}\{v_1,\cdots,v_s\}.$$

**证明** 注意到, 若 $\cos(\theta_k)=1$, 则必有 $u_k=v_k$, 从而定理成立. □

在非精确的运算下, 在算法 12.4.3 中, 有必要计算余弦值为 1 的近似重数.

**例 12.4.3** 如果

$$A=\begin{bmatrix}1&2\\3&4\\5&6\end{bmatrix},\quad B=\begin{bmatrix}1&5\\3&7\\5&-1\end{bmatrix},$$

则 $\mathrm{ran}(A)$ 和 $\mathrm{ran}(B)$ 之间的主角的余弦值为 1.000 和 0.856.

## 习题

**12.4.1** 证明: 若 $A$ 和 $B$ 是 $m\times p$ 矩阵, $p\leqslant m$, 则

$$\min_{Q^{\mathrm{T}}Q=I_p}\|A-BQ\|_F^2=\sum_{i=1}^{p}(\sigma_i(A)^2-2\sigma_i(B^{\mathrm{T}}A)+\sigma_i(B)^2).$$

**12.4.2** 推广算法 12.4.2, 使之能够计算 $\mathrm{null}(A_1)\cap\cdots\cap\mathrm{null}(A_s)$ 的标准正交基.

**12.4.3** 推广算法 12.4.3, 使之能处理 $A$ 和 $B$ 秩亏的情况.

**12.4.4** 试阐述 $\mathrm{ran}(A)$ 与 $\mathrm{ran}(B)$ 之间的主角和主向量与下述广义特征值问题的特征值和特征向量之间的关系:

$$\begin{bmatrix}0&A^{\mathrm{T}}B\\B^{\mathrm{T}}A&0\end{bmatrix}\begin{bmatrix}y\\z\end{bmatrix}=\sigma\begin{bmatrix}A^{\mathrm{T}}A&0\\0&B^{\mathrm{T}}B\end{bmatrix}\begin{bmatrix}y\\z\end{bmatrix}.$$

**12.4.5** 设 $A,B\in\mathbb{R}^{m\times n}$, 且 $A$ 列满秩. 说明如何计算一对称矩阵 $X\in\mathbb{R}^{n\times n}$ 使得 $\|AX-B\|_F$ 达到极小 (提示: 计算 $A$ 的 SVD).

## 本节注释与参考文献

在正交矩阵集上极小化 $\|A-BQ\|_F$ 出自心理测量学, 见:

B. Green (1952). "The Orthogonal Approximation of an Oblique Structure in Factor Analysis," *Psychometrika* 17, 429–440.

P. Schonemann (1966). "A Generalized Solution of the Orthogonal Procrustes Problem," *Psychometrika* 31, 1–10.

I. Y. Bar-Itzhack (1975). "Iterative Optimal Orthogonalization of the Strapdown Matrix," *IEEE Trans. Aerospace and Electronic Systems* 11, 30–37.

R. J. Hanson and M. J. Norris (1981). "Analysis of Measurements Based on the Singular Value Decomposition," *SIAM J. Sci. and Stat. Comp.* 2, 363–374.

H. Park (1991). "A Parallel Algorithm for the Unbalanced Orthogonal Procrustes Problem," *Parallel Computing* 17, 913–923.

当 $\boldsymbol{B}=\boldsymbol{I}$ 时, 这个问题变为找出离 $\boldsymbol{A}$ 最近的正交矩阵. 这也等价于 4.2.10 节的极分解问题, 见:

A. Björck and C. Bowie (1971). "An Iterative Algorithm for Computing the Best Estimate of an Orthogonal Matrix," *SIAM J. Num, Anal.* 8, 358–364.

N. J. Higham (1986). "Computing the Polar Decomposition—with Applications," *SIAM J. Sci. and Stat. Comp.* 7, 1160–1174.

如果 $\boldsymbol{A}$ 本身就比较靠近正交矩阵, 则 Björck 和 Bowie 的技巧比 SVD 更加有效. $\min\|\boldsymbol{AX}-\boldsymbol{F}\|_F$ 在约束 $\boldsymbol{X}$ 为对称矩阵的研究可见:

N. J. Higham (1988). "The Symmetric Procrustes Problem," *BIT* 28, 133–143.

用 SVD 解标准相关分析问题可见:

A. Björck and G. H. Golub (1973). "Numerical Methods for Computing Angles Between Linear Subspaces," *Math. Comp.* 27, 579–594.

G. H. Golub and H. Zha (1994). "Perturbation Analysis of the Canonical Correlations of Matrix Paris," *Lin. Alg. and Its Applic.* 210, 3–28.

SVD 在统计计算中有其他作用, 见:

S. J. Hammarling (1985). "The Singular Value Decomposition in Multivariate Statistics," *ACM SIGNUM Newsletter* 20, 2–25.

## 12.5 矩阵分解的修正

在许多应用中, 当一矩阵 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ 在某种极小意义下变化了以后, 需要重新进行分解. 例如, 假定已知 $\boldsymbol{A}$ 的 QR 分解, 但有时会需要计算它作如下变化后的 QR 分解: (a) 给 $\boldsymbol{A}$ 加上某个秩 1 矩阵. (b) 给 $\boldsymbol{A}$ 添加一行 (或列). (c) 删去 $\boldsymbol{A}$ 的一行或一列. 本节将说明在这些情况下, 修正 $\boldsymbol{A}$ 的 QR 分解比重新开始求 QR 分解要有效的多. 我们还将介绍当一矩阵增加一行后如何修正它的零空间.

在讨论开始之前, 要指出的一点是: 也有一些技术来修正分解 $\boldsymbol{PA}=\boldsymbol{LU}, \boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}, \boldsymbol{A}=\boldsymbol{LDL}^{\mathrm{T}}$. 然而, 对这些分解进行修正更为精细, 因为有时需选主元以及当处理正定矩阵时, 变化后的矩阵往往不正定. 见 Gill, Golub, Murray, and Saun-

ders(1974) 以及 Stewart(1979). 根据他们的思想, 我们简要地讨论双曲变换及其在 Cholesky 还原问题中的应用.

本节需要熟悉 3.5 节、4.1 节、5.1 节、5.2 节、5.4 节及 5.5 节中的内容. 补充阅读材料见 Gill, Murray, and Wright(1991).

### 12.5.1 秩 1 修正

假设我们已有分解 QR 分解 $\boldsymbol{QR} = \boldsymbol{B} \in \mathbb{R}^{m\times n}$, 现要求新的 QR 分解 $\boldsymbol{B} + \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}} = \boldsymbol{Q}_1\boldsymbol{R}_1$, 其中 $\boldsymbol{u} \in \mathbb{R}^m$ 和 $\boldsymbol{v} \in \mathbb{R}^n$ 为已知向量. 下式显然成立:

$$\boldsymbol{B} + \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}} = \boldsymbol{Q}(\boldsymbol{R} + \boldsymbol{w}\boldsymbol{v}^{\mathrm{T}}), \tag{12.5.1}$$

其中 $\boldsymbol{w} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{u}$. 假定已计算出旋转矩阵 $\boldsymbol{J}_{n-1}, \cdots, \boldsymbol{J}_2, \boldsymbol{J}_1$ 使得

$$\boldsymbol{J}_1^{\mathrm{T}} \cdots \boldsymbol{J}_{n-1}^{\mathrm{T}} \boldsymbol{w} = \pm \|\boldsymbol{w}\|_2 \boldsymbol{e}_1.$$

其中每一 $\boldsymbol{J}_k$ 是作用在第 $k$ 列与第 $k+1$ 列之间的旋转矩阵 (详见算法 5.1.3). 当这些 Givens 旋转矩阵作用于 $\boldsymbol{R}$ 后, 可证

$$\boldsymbol{H} = \boldsymbol{J}_1^{\mathrm{T}} \cdots \boldsymbol{J}_{n-1}^{\mathrm{T}} \boldsymbol{R} \tag{12.5.2}$$

为上 Hessenberg 矩阵. 例如当 $n = 4$ 的情形, 开始有

$$\boldsymbol{R} = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}, \quad \boldsymbol{w} = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix},$$

然后依下列次序更新:

$$\boldsymbol{R} = \boldsymbol{J}_3^{\mathrm{T}}\boldsymbol{R} = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad \boldsymbol{w} = \boldsymbol{J}_3^{\mathrm{T}}\boldsymbol{w} = \begin{bmatrix} \times \\ \times \\ \times \\ 0 \end{bmatrix},$$

$$\boldsymbol{R} = \boldsymbol{J}_2^{\mathrm{T}}\boldsymbol{R} = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad \boldsymbol{w} = \boldsymbol{J}_2^{\mathrm{T}}\boldsymbol{w} = \begin{bmatrix} \times \\ \times \\ 0 \\ 0 \end{bmatrix},$$

$$\boldsymbol{H} = \boldsymbol{J}_1^{\mathrm{T}}\boldsymbol{R} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad \boldsymbol{w} = \boldsymbol{J}_1^{\mathrm{T}}\boldsymbol{w} = \begin{bmatrix} \times \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

因此,

$$(J_1^{\mathrm{T}}\cdots J_{n-1}^{\mathrm{T}})(R+wv^{\mathrm{T}}) = H \pm \|w\|_2 e_1 v^{\mathrm{T}} = H_1 \tag{12.5.3}$$

为上 Hessenberg 矩阵.

在算法 5.2.3 中, 我们说明了如何在 $O(n^2)$ 个 flop 内计算出一个上 Hessenberg 矩阵的 QR 分解. 特别地, 我们可找出 Givens 旋转矩阵 $G_k, k=1:n-1$, 使得

$$G_{n-1}^{\mathrm{T}}\cdots G_1^{\mathrm{T}} H_1 = R_1 \tag{12.5.4}$$

为上三角形矩阵. 把 (12.5.1)~(12.5.4) 结合起来得到 QR 分解 $B+uv^{\mathrm{T}} = Q_1R_1$, 其中

$$Q_1 = QJ_{n-1}\cdots J_1G_1\cdots G_{n-1}.$$

仔细的分析可知这需要约 $26n^2$ 个 flop. 计算向量 $w=Q^{\mathrm{T}}u$ 要 $2n^2$ 个 flop, 计算 $H$ 并把 $J_k$ 乘到 $Q$ 上需 $12n^2$ 个 flop. 最后, 计算 $R_1$ 并把 $G_k$ 乘到 $Q$ 上需 $12n^2$ 个 flop.

这一技巧对 $B$ 为长方形矩阵的情形也是适用的. 它也可推广到计算 $B+UV^{\mathrm{T}}$ 的 QR 分解. 其中 $\mathrm{rank}(UV^{\mathrm{T}}) = p > 1$.

### 12.5.2 增加或删去一列的情况

假设已有 QR 分解:

$$QR = A = [a_1, \cdots, a_n], \quad a_i \in \mathbb{R}^m, \tag{12.5.5}$$

把上三角形矩阵 $R\in\mathbb{R}^{m\times n}$ 作如下分块:

$$R = \begin{bmatrix} R_{11} & v & R_{13} \\ 0 & r_{kk} & w^{\mathrm{T}} \\ 0 & 0 & R_{33} \end{bmatrix} \begin{matrix} k-1 \\ 1 \\ m-k \end{matrix}.$$
$$\qquad\qquad k-1 \quad 1 \quad n-k$$

下面我们要计算下述矩阵的 QR 分解:

$$\tilde{A} = [a_1, \cdots, a_{k-1}, a_{k+1}, \cdots, a_n] \in \mathbb{R}^{m\times(n-1)}.$$

注意 $\tilde{A}$ 是由 $A$ 中删去第 $k$ 列所得, 且

$$Q^{\mathrm{T}}\tilde{A} = \begin{bmatrix} R_{11} & R_{13} \\ 0 & w^{\mathrm{T}} \\ 0 & R_{33} \end{bmatrix} = H$$

为上 Hessenberg 矩阵. 例如:

$$
\boldsymbol{H}=\begin{bmatrix}
\times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times \\
0 & 0 & \times & \times & \times \\
0 & 0 & 0 & \times & \times \\
0 & 0 & 0 & 0 & \times \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}, \quad m=7, \quad n=6, \quad k=3.
$$

显然, 不需要的次对角元 $h_{k+1,k},\cdots,h_{n,n-1}$ 可用一系列的 Givens 旋转化为零: $\boldsymbol{G}_{n-1}^{\mathrm{T}}\cdots\boldsymbol{G}_{k}^{\mathrm{T}}\boldsymbol{H}=\boldsymbol{R}_1$. 这里 $\boldsymbol{G}_i$ 是作用在第 $i$ 列与第 $i+1$ 列之间的旋转, $i=k:n-1$. 因此, 若 $\boldsymbol{Q}_1=\boldsymbol{Q}\boldsymbol{G}_k\cdots\boldsymbol{G}_{n-1}$, 则 $\tilde{\boldsymbol{A}}=\boldsymbol{Q}_1\boldsymbol{R}_1$ 为 $\tilde{\boldsymbol{A}}$ 的 QR 分解.

上面的修正程序可在 $O(n^2)$ 个 flop 完成, 它在某些特定的最小二乘问题中很有用. 例如, 为了检验基本模型中第 $k$ 个因素的重要性, 可以在相应的数据矩阵中删去此列, 并解所得矩阵的 LS 问题.

同样, 能够有效地计算矩阵 $\boldsymbol{A}$ 增加一列后相应的 LS 问题的解也是很有用处的. 假设已有 QR 分解 (12.5.5), 现要计算

$$
\tilde{\boldsymbol{A}}=[\boldsymbol{a}_1,\cdots,\boldsymbol{a}_k,\boldsymbol{z},\boldsymbol{a}_{k+1},\cdots,\boldsymbol{a}_n]
$$

的 QR 分解, 其中 $\boldsymbol{z}\in\mathbb{R}^m$ 已知. 若 $\boldsymbol{w}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{z}$, 则

$$
\boldsymbol{Q}^T\boldsymbol{A}=[\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{a}_1,\cdots,\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{a}_k,\boldsymbol{w},\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{a}_{k+1},\cdots,\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{a}_n]=\tilde{\boldsymbol{A}},
$$

除了第 $k+1$ 列的 “尖峰” 外, 它是上三角形的. 例如:

$$
\tilde{\boldsymbol{A}}=\begin{bmatrix}
\times & \times & \times & \times & \times & \times \\
0 & \times & \times & \times & \times & \times \\
0 & 0 & \times & \times & \times & \times \\
0 & 0 & 0 & \times & \times & \times \\
0 & 0 & 0 & \times & 0 & \times \\
0 & 0 & 0 & \times & 0 & 0 \\
0 & 0 & 0 & \times & 0 & 0
\end{bmatrix}, \quad m=7, \quad n=5, \quad k=3
$$

可以找出 Givens 旋转 $\boldsymbol{J}_{m-1},\cdots,\boldsymbol{J}_{k+1}$ 使得

$$
\boldsymbol{J}_{k+1}^{\mathrm{T}}\cdots\boldsymbol{J}_{m-1}^{\mathrm{T}}\boldsymbol{w}=\begin{bmatrix}
w_1 \\ \vdots \\ w_{k+1} \\ 0 \\ \vdots \\ 0
\end{bmatrix},
$$

且 $J_{k+1}^{\mathrm{T}}\cdots J_{m-1}^{\mathrm{T}}\tilde{A}=\tilde{R}$ 为上三角形矩阵. 我们继续用上面的例子来说明这一点.

$$H=J_6^{\mathrm{T}}\tilde{A}=\begin{bmatrix}\times&\times&\times&\times&\times&\times\\0&\times&\times&\times&\times&\times\\0&0&\times&\times&\times&\times\\0&0&0&\times&\times&\times\\0&0&0&\times&0&\times\\0&0&0&\times&0&0\\0&0&0&0&0&0\end{bmatrix},$$

$$H=J_5^{\mathrm{T}}H=\begin{bmatrix}\times&\times&\times&\times&\times&\times\\0&\times&\times&\times&\times&\times\\0&0&\times&\times&\times&\times\\0&0&0&\times&\times&\times\\0&0&0&\times&0&\times\\0&0&0&0&0&\times\\0&0&0&0&0&0\end{bmatrix},$$

$$H=J_4^{\mathrm{T}}H=\begin{bmatrix}\times&\times&\times&\times&\times&\times\\0&\times&\times&\times&\times&\times\\0&0&\times&\times&\times&\times\\0&0&0&\times&\times&\times\\0&0&0&0&\times&\times\\0&0&0&0&0&\times\\0&0&0&0&0&0\end{bmatrix},$$

此修正大约需要 $O(mn)$ 个 flop.

### 12.5.3 增加或删去一行

假设已知 QR 分解$QR=A\in\mathbb{R}^{m\times n}$, 现要求 $\tilde{A}=\begin{bmatrix}w^{\mathrm{T}}\\A\end{bmatrix}$ 的 QR 分解, 其中 $w\in\mathbb{R}^n$. 注意, $\mathrm{diag}(1,Q^{\mathrm{T}})\tilde{A}=\begin{bmatrix}w^{\mathrm{T}}\\R\end{bmatrix}=H$, $H$ 为上 Hessenberg 矩阵. 因此, 可找出 Givens 旋转矩阵 $J_1\cdots,J_n$ 使得 $J_n^{\mathrm{T}}\cdots J_1^{\mathrm{T}}H=R_1$ 为上三角形矩阵. 由此可知 $\tilde{A}=Q_1R_1$ 便是所要的 QR 分解, 其中 $Q_1=\mathrm{diag}(1,Q)J_1\cdots J_n$.

如果在 $A$ 的第 $k$ 行与第 $k+1$ 行之间增加新的一行, 修正分解没有本质上的

困难. 只要在上面的步骤中以 $\boldsymbol{PA}$ 代替 $\boldsymbol{A}$, 以 $\boldsymbol{PQ}$ 代替 $\boldsymbol{Q}$, 其中

$$\boldsymbol{P}=\begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{m-k} \\ \boldsymbol{I}_k & \mathbf{0} \end{bmatrix}.$$

结束后, $\mathrm{diag}(1,\boldsymbol{P}^{\mathrm{T}})\boldsymbol{Q}_1$ 就是所要的正交分解因子.

最后, 我们考虑一下当 $\boldsymbol{A}$ 的第一行删去时, 如何修正它的 QR 分解 $\boldsymbol{QR}=\boldsymbol{A}\in\mathbb{R}^{m\times n}$. 确切地说, 我们希望计算出子矩阵 $\boldsymbol{A}_1$ 的 QR 分解, 这里

$$\boldsymbol{A}=\begin{bmatrix} \boldsymbol{z}^{\mathrm{T}} \\ \boldsymbol{A}_1 \end{bmatrix}\begin{matrix} 1 \\ m-1 \end{matrix}$$

(删去任意一列的情形是类似的). 令 $\boldsymbol{q}^{\mathrm{T}}$ 为 $\boldsymbol{Q}$ 的第一行, 且计算出 Givens 旋转矩阵 $\boldsymbol{G}_1,\cdots,\boldsymbol{G}_{m-1}$ 使得

$$\boldsymbol{G}_1^{\mathrm{T}}\cdots\boldsymbol{G}_{m-1}^{\mathrm{T}}\boldsymbol{q}=\alpha\boldsymbol{e}_1,\quad \alpha=\pm1.$$

注意

$$\boldsymbol{H}=\boldsymbol{G}_1^{\mathrm{T}}\cdots\boldsymbol{G}_{m-1}^{\mathrm{T}}\boldsymbol{R}=\begin{bmatrix} \boldsymbol{v}^{\mathrm{T}} \\ \boldsymbol{R}_1 \end{bmatrix}\begin{matrix} 1 \\ m-1 \end{matrix}$$

是上 Hessenberg 矩阵且

$$\boldsymbol{QG}_{m-1}\cdots\boldsymbol{G}_1=\begin{bmatrix} \alpha & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_1 \end{bmatrix},$$

其中 $\boldsymbol{Q}_1\in\mathbb{R}^{(m-1)\times(m-1)}$ 是正交的. 因此,

$$\boldsymbol{A}=\begin{bmatrix} \boldsymbol{z}^{\mathrm{T}} \\ \boldsymbol{A}_1 \end{bmatrix}=(\boldsymbol{QG}_{m-1}\cdots\boldsymbol{G}_1)(\boldsymbol{G}_1^{\mathrm{T}}\cdots\boldsymbol{G}_{m-1}^{\mathrm{T}}\boldsymbol{R})=\begin{bmatrix} \alpha & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_1 \end{bmatrix}\begin{bmatrix} \boldsymbol{v}^{\mathrm{T}} \\ \boldsymbol{R}_1 \end{bmatrix},$$

从中可以看出 $\boldsymbol{A}_1=\boldsymbol{Q}_1\boldsymbol{R}_1$ 就是所要的 QR 分解.

### 12.5.4 双曲变换法

我们曾提到过分解 $\boldsymbol{A}=\boldsymbol{QR}$ 中的 "$\boldsymbol{R}$" 是 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{GG}^{\mathrm{T}}$ 中的转置 Cholesky 因子. 因此, 刚刚讨论过的 QR 修正算法与 Cholesky 分解类似的修正问题之间有非常紧密的联系. 我们以 Cholesky *还原问题* 来说明这一点, 它相应于在 QR 分解中删去矩阵 $\boldsymbol{A}$ 的一行的情形. 在这一问题中, 已有 Cholesky 分解:

$$\boldsymbol{GG}^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix} \boldsymbol{z}^{\mathrm{T}} \\ \boldsymbol{A}_1 \end{bmatrix}^{\mathrm{T}}\begin{bmatrix} \boldsymbol{z}^{\mathrm{T}} \\ \boldsymbol{A}_1 \end{bmatrix}, \tag{12.5.6}$$

其中 $\boldsymbol{A}\in\mathbb{R}^{m\times n}, m>n, z\in\mathbb{R}^n$. 我们的目标是寻找一低阶的三角形矩阵 $\boldsymbol{G}_1$ 使得 $\boldsymbol{G}_1\boldsymbol{G}_1^{\mathrm{T}}=\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1$. 对这个有趣而重要的问题有好几种不同的途径来处理. 现给出一个依赖于*双曲变换*的还原步骤, 它可以引进一些新的思想.

我们从一个概念开始. 矩阵 $\boldsymbol{H} \in \mathbb{R}^{m\times m}$ 称为关于**符号矩阵** $\boldsymbol{S} = \mathrm{diag}(\pm 1)$ 是**伪正交的**, 若 $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{H} = \boldsymbol{S}$. 从 (12.5.6) 有 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} = \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1 + \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$, 于是有

$$\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1 = \boldsymbol{A}^T\boldsymbol{A} - \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} = \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}} - \boldsymbol{z}\boldsymbol{z}^{\mathrm{T}} = [\boldsymbol{G} \quad \boldsymbol{z}]\begin{bmatrix} \boldsymbol{I}_n & 0 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} \boldsymbol{G}^{\mathrm{T}} \\ \boldsymbol{z}^{\mathrm{T}} \end{bmatrix}.$$

定义符号矩阵

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{I}_n & \mathbf{0} \\ \mathbf{0} & -1 \end{bmatrix}, \tag{12.5.7}$$

假定能找到矩阵 $\boldsymbol{H} \in \mathbb{R}^{(n+1)\times(n+1)}$ 使得 $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{H} = \boldsymbol{S}$, 且有性质:

$$\boldsymbol{H}\begin{bmatrix} \boldsymbol{G}^{\mathrm{T}} \\ \boldsymbol{z}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{G}_1^{\mathrm{T}} \\ \mathbf{0} \end{bmatrix} \tag{12.5.8}$$

为上三角形矩阵. 由此可得:

$$\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{A}_1 = [\boldsymbol{G} \quad \boldsymbol{z}]\boldsymbol{H}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{H}\begin{bmatrix} \boldsymbol{G}^{\mathrm{T}} \\ \boldsymbol{z}^{\mathrm{T}} \end{bmatrix} = [\boldsymbol{G}_1 \quad \mathbf{0}]\boldsymbol{S}\begin{bmatrix} \boldsymbol{G}_1 \\ \mathbf{0} \end{bmatrix} = \boldsymbol{G}_1\boldsymbol{G}_1^{\mathrm{T}}.$$

这就是所寻找的 Cholesky 分解.

现在我们说明在 (12.5.8) 中怎样利用**双曲旋转**构造双曲变换矩阵, 一个 $2\times 2$ 的双曲旋转具有如下形式:

$$\boldsymbol{H} = \begin{bmatrix} \cosh(\theta) & -\sinh(\theta) \\ -\sinh(\theta) & \cosh(\theta) \end{bmatrix} = \begin{bmatrix} c & -s \\ -s & c \end{bmatrix}.$$

注意, 如果 $\boldsymbol{H} \in \mathbb{R}^{2\times 2}$ 是一双曲旋转矩阵, 则 $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{H} = \boldsymbol{S}$, 其中 $\boldsymbol{S} = \mathrm{diag}(-1, 1)$. 和 Givens 旋转矩阵一样, 双曲旋转矩阵也可用来消去元素. 从

$$\begin{bmatrix} c & -s \\ -s & c \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}, \quad c^2 - s^2 = 1$$

可得方程 $cx_2 = sx_1$. 注意到当 $x_1 = x_2 \neq 0$ 时, 此方程无解, 双曲旋转矩阵在数值上不如 Givens 旋转矩阵那么稳固. 若 $x_1 \neq x_2$, 则可计算出相应的 cosh-sinh 对:

**if** $x_2 = 0$

  $s = 0;\ c = 1$

**else**

  **if** $|x_2| < |x_1|$

    $\tau = x_2/x_1;\ c = 1/\sqrt{1-\tau^2};\ s = c\tau$ (12.5.9)

  **elseif** $|x_1| < |x_2|$

    $\tau = x_1/x_2;\ s = 1/\sqrt{1-\tau^2};\ c = s\tau$

  **end**

**end**

观察可知，这一算法产生的双曲旋转矩阵的范数随着 $x_1$ 趋近于 $x_2$ 而变得越来越大.

设矩阵 $\boldsymbol{H} = H(p, n+1, \theta) \in \mathbb{R}^{(n+1)\times(n+1)}$ 除了元素 $h_{pp} = h_{n+1,n+1} = \cosh(\theta)$, $h_{p,n+1} = h_{n+1,p} = -\sinh(\theta)$ 外, 其余元素都是 1, 则必有 $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{H} = \boldsymbol{S}$, 其中 $\boldsymbol{S}$ 如 (12.5.7) 所述. 利用 (12.5.9), 我们试图对 $k = 2 : n+1$ 找出双曲旋转矩阵 $\boldsymbol{H}_k = H(1, k, \theta_k)$ 使得

$$\boldsymbol{H}_n \cdots \boldsymbol{H}_1 \begin{bmatrix} \boldsymbol{G}^{\mathrm{T}} \\ \boldsymbol{z}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{G}}^{\mathrm{T}} \\ \boldsymbol{0} \end{bmatrix}.$$

如果 $\boldsymbol{A}$ 是列满秩的, 这是可以办到的. 双曲旋转将元素 $(k+1, k)$ 的值化为零. 换句话说, 若 $\boldsymbol{A}$ 列满秩, 则可证每次调用 (12.5.9) 必可产生一 cosh-sinh 对, 见 Alexander, Pan, and Plemmons(1988).

### 12.5.5 修正 ULV 分解

假设 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ 是秩亏的, 且有它的零空间的一组基. 如果给 $\boldsymbol{A}$ 增加一行,

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{z}^{\mathrm{T}} \end{bmatrix},$$

怎样容易地求出 $\tilde{\boldsymbol{A}}$ 的零空间的基呢? 当涉及一系列这样的修正问题时, 就变成了跟踪零空间的问题. 子空间的跟踪问题出现在很多实时信号处理的应用中.

用 SVD 来计算是很笨的方法, 这是因为重新计算秩 1 扰动后的矩阵的 SVD 需 $O(n^3)$ 个 flop. 不过, Stewart(1993) 证明了, 如果把 3.5.4 节中条件数估计的思想与完全正交分解结合起来, 则零空间的修正问题可用 $O(n^2)$ 个 flop 来完成. 回想 5.4.2 节的内容便可知, 完全正交分解是从两边进行的, 且可显示原矩阵的秩:

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \begin{bmatrix} \boldsymbol{T}_{11} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{T}_{11} \in \mathbb{R}^{r\times r}, \quad r = \mathrm{rank}(\boldsymbol{A}).$$

可用一对 QR 分解 (其中一个用到了列选主) 来实现这一点. 在这种情况下, 精确的运算可使 $\boldsymbol{T}_{11} = \boldsymbol{L}$ 为下三角形矩阵. 但由于噪声和舍入误差的影响, 我们实际计算下面的形式:

$$\boldsymbol{U}^{T}\boldsymbol{A}\boldsymbol{V} = \begin{bmatrix} \boldsymbol{L} & \boldsymbol{0} \\ \boldsymbol{H} & \boldsymbol{E} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \tag{12.5.10}$$

其中 $\boldsymbol{L} \in \mathbb{R}^{r\times r}$ 与 $\boldsymbol{E} \in \mathbb{R}^{(n-r)\times(n-r)}$ 为下三角形矩阵. 和 $\sigma_{\min}(\boldsymbol{L})$ 相比, $\boldsymbol{H}$ 和 $\boldsymbol{E}$ 是小的矩阵. 在这种情况下, 我们把 (12.5.10) 称为**显秩 $\boldsymbol{ULV}$ 分解**. 若作划分

$$\boldsymbol{V} = [\underset{r}{\boldsymbol{V}_1} \quad \underset{n-r}{\boldsymbol{V}_2}], \qquad \boldsymbol{U} = [\underset{r}{\boldsymbol{U}_1} \quad \underset{m-r}{\boldsymbol{U}_2}],$$

则 $V_2$ 的列定义了一近似零空间：

$$\|AV_2\|_2 = \|U_2E\|_2 \leqslant \|E\|_2.$$

我们的目标是计算出增加了一行的矩阵 $\tilde{A}$ 的显秩 ULV 分解[①]. 更明确地讲, 我们的任务是如何修正矩阵 $L, E, H, V$ 和秩数 (可能的话), 且在 $O(n^2)$ 个 flop 完成.

注意到

$$\begin{bmatrix} U & 0 \\ 0 & 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} A \\ z^{\mathrm{T}} \end{bmatrix} V = \begin{bmatrix} L & 0 \\ H & E \\ 0 & 0 \\ w^{\mathrm{T}} & y^{\mathrm{T}} \end{bmatrix}.$$

交换最后一行和矩阵 $H$ 与 $E$ 下面的零行, 可以看到, 问题在于用 $O(n^2)$ 个 flop 计算

$$\begin{bmatrix} L & 0 \\ H & E \\ w^{\mathrm{T}} & y^{\mathrm{T}} \end{bmatrix} = \left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & 0 & 0 & 0 \\ l & l & l & l & 0 & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline w & w & w & w & y & y & y \end{array}\right] \tag{12.5.11}$$

的显秩 ULV 分解. 这里, 以 $r=4$ 和 $n=7$ 为例来说明主要思想. 记住, 元素 $h$ 和 $e$ 是很小的, 且已经导出数值秩是 4. 在实际中, 这涉及与 5.5.7 节中较小的容许值比较的问题.

应用与 12.5.3 节中类似的消元技巧, 可用一系列的行旋转把最后一行化为零元:

$$\begin{bmatrix} \tilde{L} \\ \hline 0 \end{bmatrix} = \left[\begin{array}{cccc|ccc} \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 \\ \hline \times & \times & \times & \times & \times & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 \\ \times & \times & \times & \times & \times & \times & \times \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

① 与此对称的是显秩 URV 分解, 在有些情形用 URV 形式比 ULV 更好.

因为这种化零过程把最后一行的元素 (可能很大) 与其他行的元素混和在一起, 所以三角部分一般不是显秩的. 然而, 可以联合利用条件数估计和旋转化零的技巧来恢复显秩的结构. 假设增加一行后, 零空间的维数为 2.

用一可靠的条件数估计量, 我们可得一 2 范数单位向量 $p$, 使得

$$\|\boldsymbol{p}^{\mathrm{T}}\tilde{\boldsymbol{L}}\|_2 \approx \sigma_{\min}(\tilde{\boldsymbol{L}}).$$

见 3.5.4 节. 可以找到旋转矩阵 $\{\boldsymbol{U}_{i,i+1}\}_{i=1}^{6}$ 使得

$$\boldsymbol{U}_{67}^{\mathrm{T}}\boldsymbol{U}_{56}^{\mathrm{T}}\boldsymbol{U}_{45}^{\mathrm{T}}\boldsymbol{U}_{34}^{\mathrm{T}}\boldsymbol{U}_{23}^{\mathrm{T}}\boldsymbol{U}_{12}^{\mathrm{T}}p = e_8 = \boldsymbol{I}_8(:,8).$$

矩阵 $\boldsymbol{H} = \boldsymbol{U}_{67}^{\mathrm{T}}\boldsymbol{U}_{56}^{\mathrm{T}}\boldsymbol{U}_{45}^{\mathrm{T}}\boldsymbol{U}_{34}^{\mathrm{T}}\boldsymbol{U}_{23}^{\mathrm{T}}\boldsymbol{U}_{12}^{\mathrm{T}}\tilde{\boldsymbol{L}}$ 为下 Hessenberg 矩阵, 它可用一系列的列旋转恢复为下三角形式 $\boldsymbol{L}_+$:

$$\boldsymbol{L}_+ = \boldsymbol{H}\boldsymbol{V}_{12}\boldsymbol{V}_{23}\boldsymbol{V}_{34}\boldsymbol{V}_{45}\boldsymbol{V}_{56}\boldsymbol{V}_{67}.$$

可得,

$$e_8^{\mathrm{T}}\boldsymbol{L}_+ = (e_8^{\mathrm{T}}\boldsymbol{H})\boldsymbol{V}_{12}\boldsymbol{V}_{23}\boldsymbol{V}_{34}\boldsymbol{V}_{45}\boldsymbol{V}_{56}\boldsymbol{V}_{67} = (p^{\mathrm{T}}\tilde{\boldsymbol{L}})\boldsymbol{V}_{12}\boldsymbol{V}_{23}\boldsymbol{V}_{34}\boldsymbol{V}_{45}\boldsymbol{V}_{56}\boldsymbol{V}_{67}$$

的范数近似值为 $\sigma_{\min}(\tilde{\boldsymbol{L}})$. 因此, 我们得到一个具有如下形式的下三角形矩阵:

$$\left[\begin{array}{cccccc|c} \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 \\ \hline h & h & h & h & h & h & e \end{array}\right],$$

其中, $h$ 和 $e$ 很小. 我们可以对上述矩阵中的 6×6 主子矩阵重复条件数估计和赶零的技巧, 或许又能得到另一元素很小的行:

$$\left[\begin{array}{ccccc|cc} \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 \\ \hline h & h & h & h & h & e & 0 \\ h & h & h & h & h & e & e \end{array}\right]$$

(否则, 说明秩数为 6). 继续重复这一过程, 可以把任何下三角形矩阵变成显秩的形式.

在 (12.5.11) 中向量 $\boldsymbol{y}$ 很小的情况下, 我们可以通过一种不同的更加有效的途径来化成显秩的形式. 我们开始用一系列左 Givens 旋转和右 Givens 旋转把 $\boldsymbol{y}$ 除第一个分量外的所有分量都化为零:

$$
\xrightarrow{\boldsymbol{V}_{67}}
\left[\begin{array}{cccc|ccc}
l & 0 & 0 & 0 & 0 & 0 & 0 \\
l & l & 0 & 0 & 0 & 0 & 0 \\
l & l & l & 0 & 0 & 0 & 0 \\
l & l & l & l & 0 & 0 & 0 \\ \hline
h & h & h & h & e & 0 & 0 \\
h & h & h & h & e & e & e \\
h & h & h & h & e & e & e \\ \hline
x & x & x & x & y & y & 0
\end{array}\right]
\xrightarrow{\boldsymbol{U}_{67}}
\left[\begin{array}{cccc|ccc}
l & 0 & 0 & 0 & 0 & 0 & 0 \\
l & l & 0 & 0 & 0 & 0 & 0 \\
l & l & l & 0 & 0 & 0 & 0 \\
l & l & l & l & 0 & 0 & 0 \\ \hline
h & h & h & h & e & 0 & 0 \\
h & h & h & h & e & e & 0 \\
h & h & h & h & e & e & e \\ \hline
x & x & x & x & y & y & 0
\end{array}\right]
$$

$$
\xrightarrow{\boldsymbol{V}_{56}}
\left[\begin{array}{cccc|ccc}
l & 0 & 0 & 0 & 0 & 0 & 0 \\
l & l & 0 & 0 & 0 & 0 & 0 \\
l & l & l & 0 & 0 & 0 & 0 \\
l & l & l & l & 0 & 0 & 0 \\ \hline
h & h & h & h & e & e & 0 \\
h & h & h & h & e & e & 0 \\
h & h & h & h & e & e & e \\ \hline
x & x & x & x & y_* & 0 & 0
\end{array}\right]
\xrightarrow{\boldsymbol{U}_{56}}
\left[\begin{array}{cccc|ccc}
l & 0 & 0 & 0 & 0 & 0 & 0 \\
l & l & 0 & 0 & 0 & 0 & 0 \\
l & l & l & 0 & 0 & 0 & 0 \\
l & l & l & l & 0 & 0 & 0 \\ \hline
h & h & h & h & e & 0 & 0 \\
h & h & h & h & e & e & 0 \\
h & h & h & h & e & e & e \\ \hline
x & x & x & x & y_* & 0 & 0
\end{array}\right],
$$

这里 "$\boldsymbol{U}_{ij}$" 指第 $i$ 行与第 $j$ 行之间的旋转矩阵, "$\boldsymbol{V}_{ij}$" 指第 $i$ 列与第 $j$ 列之间的旋转矩阵. 注意, 在这一过程中大数和小数没有混合在一起运算, 这一点很重要. 元素 $h$ 和 $e$ 还是很小.

按照这一过程, 我们可以找到一系列旋转变换阵把矩阵化成如下形式:

$$
\left[\begin{array}{cccc|ccc}
l & 0 & 0 & 0 & 0 & 0 & 0 \\
l & l & 0 & 0 & 0 & 0 & 0 \\
l & l & l & 0 & 0 & 0 & 0 \\
l & l & l & l & 0 & 0 & 0 \\ \hline
h & h & h & h & e & 0 & 0 \\
h & h & h & h & e & e & 0 \\
h & h & h & h & e & e & e \\ \hline
y & y & y & y & y & 0 & 0
\end{array}\right], \tag{12.5.12}
$$

其中元素 $y$ 很小:

$$\xrightarrow{U_{48}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & 0 & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & e \\ h & h & h & h & e & e & e \\ \hline x & x & x & 0 & y & 0 & 0 \end{array}\right] \xrightarrow{U_{38}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & \mu & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline x & x & 0 & 0 & y & 0 & 0 \end{array}\right]$$

$$\xrightarrow{U_{28}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & \mu & 0 & 0 \\ l & l & l & 0 & \mu & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline x & 0 & 0 & 0 & y & 0 & 0 \end{array}\right] \xrightarrow{U_{18}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & \mu & 0 & 0 \\ l & l & 0 & 0 & \mu & 0 & 0 \\ l & l & l & 0 & \mu & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline 0 & 0 & 0 & 0 & y_{**} & 0 & 0 \end{array}\right].$$

注意, $y_{**}$ 也是比较小的, 因为 2 范数在旋转变换下保持不变. (1, 5), (2, 5), (3, 5) 和 (4, 5) 位置上的列旋转变换可把元素 $\mu$ 化为零:

$$\xrightarrow{V_{15}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & \mu & 0 & 0 \\ l & l & l & 0 & \mu & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline y & 0 & 0 & 0 & y & 0 & 0 \end{array}\right] \xrightarrow{V_{25}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & \mu & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline y & y & 0 & 0 & y & 0 & 0 \end{array}\right]$$

$$\xrightarrow{V_{35}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & 0 & 0 & 0 \\ l & l & l & l & \mu & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline y & y & y & 0 & y & 0 & 0 \end{array}\right] \xrightarrow{V_{45}}\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & 0 & 0 & 0 \\ l & l & l & l & 0 & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ \hline y & y & y & y & y & 0 & 0 \end{array}\right]$$

因此, 这就化成了 (12.5.12) 中的结构. 所有元素 $y$ 都是很小的, 所以可用一系列的行旋转变换 $\boldsymbol{U}_{57}, \boldsymbol{U}_{47}, \cdots, \boldsymbol{U}_{17}$ 把最后一行的元素化成零, 得到显秩的形式:

$$\left[\begin{array}{cccc|ccc} l & 0 & 0 & 0 & 0 & 0 & 0 \\ l & l & 0 & 0 & 0 & 0 & 0 \\ l & l & l & 0 & 0 & 0 & 0 \\ l & l & l & l & 0 & 0 & 0 \\ \hline h & h & h & h & e & 0 & 0 \\ h & h & h & h & e & e & 0 \\ h & h & h & h & e & e & e \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

## 习　　题

**12.5.1**　假设已有 $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ 的 QR 分解, 现要求极小化问题 $\min \|(\boldsymbol{A}+\boldsymbol{u}\boldsymbol{v}^{\mathrm{T}})\boldsymbol{x}-\boldsymbol{b}\|_2$ 的解, 其中 $\boldsymbol{u}, \boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{v} \in \mathbb{R}^n$ 给定. 给出一个求解此问题且只需 $O(mn)$ 个 flop 的算法. 假定必须修正 $\boldsymbol{Q}$.

**12.5.2**　现有 QR 分解 $\boldsymbol{QR} = \boldsymbol{A} \in \mathbb{R}^{m \times n}$. 给出一个求去掉 $\boldsymbol{A}$ 的第 $k$ 行所剩下的矩阵的 QR 分解算法, 它只需 $O(mn)$ 个 flop.

**12.5.3**　设 $\boldsymbol{T} \in \mathbb{R}^{n \times n}$ 是三对角对称矩阵. $\boldsymbol{v} \in \mathbb{R}^n$, 说明怎样用 Lanczos 算法, 在 $O(n^2)$ 个 flop 内计算正交矩阵 $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{T}+\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}})\boldsymbol{Q} = \tilde{\boldsymbol{T}}$ 为三对角的.

**12.5.4**　设 $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{c}^{\mathrm{T}} \\ \boldsymbol{B} \end{bmatrix}$ 列满秩, $\boldsymbol{c} \in \mathbb{R}^n$, $\boldsymbol{B} \in \mathbb{R}^{(m-1)\times n}$, $m > n$, 用 Sherman-Morrison-Woodbury 公式证明:

$$\frac{1}{\sigma_{\min}(\boldsymbol{B})} \leqslant \frac{1}{\sigma_{\min}(\boldsymbol{A})} + \frac{\|(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{c}\|_2^2}{1-\boldsymbol{c}^{\mathrm{T}}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{c}}$$

**12.5.5**　作为 $x_1$ 和 $x_2$ 的函数, (12.5.9) 产生的双曲旋转矩阵的 2 范数是多少?

**12.5.6**　证明: 如果 $\boldsymbol{A}$ 列满秩, 则 12.5.4 节的双曲约化不会失败.

**12.5.7**　设 $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{H} \\ \boldsymbol{0} & \boldsymbol{E} \end{bmatrix}$, 这里 $\boldsymbol{R}$ 和 $\boldsymbol{E}$ 都是方阵且 $\rho = \dfrac{\|\boldsymbol{E}\|_2}{\sigma_{\min}(\boldsymbol{R})} < 1$. 证明: 若 $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}_{11} & \boldsymbol{Q}_{12} \\ \boldsymbol{Q}_{21} & \boldsymbol{Q}_{22} \end{bmatrix}$ 正交, 且

$$\begin{bmatrix} \boldsymbol{R} & \boldsymbol{H} \\ \boldsymbol{0} & \boldsymbol{E} \end{bmatrix}\begin{bmatrix} \boldsymbol{Q}_{11} & \boldsymbol{Q}_{12} \\ \boldsymbol{Q}_{21} & \boldsymbol{Q}_{22} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1 & \boldsymbol{0} \\ \boldsymbol{H}_1 & \boldsymbol{E}_1 \end{bmatrix}$$

则 $\|\boldsymbol{H}_1\|_2 \leqslant \rho\|\boldsymbol{H}\|_2$.

## 本节注释与参考文献

修正问题的诸多方面可见:

P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders (1974). "Methods for Modifying Matrix Factorizations," *Math. Comp.* 28, 505–535.

优化中的应用可见:

R. H. Bartels (1971). "A Stabilization of the Simplex Method," *Numer. Math.* 16, 414–434.

P. E. Gill, W. Murray, and M. A. Saunders (1975). "Methods for Computing and Modifying the LDV Factors of a Matrix," *Math. Comp.* 29, 1051–1077.

D. Goldfarb (1976). "Factored Variable Metric Methods for Unconstrained Optimization," *Math. Comp.* 30. 796–811.

J. E. Dennis and R. B. Schnabel (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ.

W. W. Hager (1989). "Updating the Inverse of a Matrix," *SIAM Review* 31, 221–239.

S. K. Eldersveld and M. A. Saunders (1992). "A Block-LU Update for Large-Scale Linear Programming," *SIAM J. Matrix Anal. Appl.* 13, 191–201.

在最小二乘中的修正问题的讨论见:

J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart (1976). "Reorthogonaization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization," *Math. Comp.* 30, 772–795.

S. Qiao (1988). "Recursive Least Squares Algorithm for Linear Prediction Problems," *SIAM J. Matrix Anal. Appl.* 9, 323–328.

Å. Björck, H. Park, and L. Eldén (1994). "Accurate Downdating of Least Squares Solutions," *SIAM J. Matrix Anal. Appl.* 15, 549–568.

S. J. Olszanskyj, J. M. Lebak, and A. W. Bojanczyk (1994). "Rank-k Modification Methods for Recursive Least Squares Problems," *Numerical Algorithms* 7, 325–354.

L. Eldén and H. Park (1994). "Block Downdating of Least Squares Solutions," *SIAM J. Matrix Anal. Appl.* 15, 1018–1034.

另一个重要的课题是条件数估计的修正:

W. R. Ferng, G. H. Golub, and R. J. Plemmons (1991). "Adaptive Lanczos Methods for Recursive Condition Estimation," *Numerical Algorithms* 1, 1–20.

G. Shroff and C. H. Bischof (1992). "Adaptive Condition Estimation for Rank-One Updates of QR Factorizations," *SIAM J. Matrix Anal. Appl.* 13, 1264–1278.

D. J. Pierce and R. J. Plemmons (1992). "Fast Adaptive Condition Estimation," *SIAM J. Matrix Anal. Appl.* 13, 274–291.

双曲变换的讨论可见:

G. H. Golub (1969). "Matrix Decompositions and Statistical Computation," in *Statistical Computation,* ed., R. C. Milton and J. A. Nelder, Academic Press, New York, pp. 365–397.

C. M. Rader and A. O. Steinhardt (1988). "Hyperbolic Householder Transforms," *SIMA J. Matrix Anal. Appl.* 9, 269–290.

S. T. Alexander, C. T. Pan, and R. J. Plemmons (1988). "Analysis of a Recursive Least Squares Hyperbolic Rotation Algorithm for Signal Processing," *Lin. Alg. and Its Applic.*

98, 3–40.

G. Cybenko and M. Berry (1990). "Hyperbolic Householder Algorithms for Factoring Structured Matrices," *SIAM J. Matrix Anal. Appl.* 11, 499–520.

A. W. Bojanczyk, R. Onn, and A. O. Steinhardt (1993). "Existence of the Hyperbolic Singular Value Decomposition," *Lin, Alg, and Its Applic.* 185, 21–30.

Cholesky 分解的修正也吸引了不少注意力, 见:

G. W. Stewart (1979). "The Effects of Rounding Error on an Algorithm for Downdating a Cholesky Factorization," *J. Inst. Math. Applic.* 23, 203–213.

A. W. Bojanczyk, R. P. Brent, P. Van Dooren, and F. R. de Hoog (1987). "A Note on Downdating the Cholesky Factorization," *SIAM J. Sci. and Stat. Comp.* 8, 210–221.

C. S. Henkel, M. T. Heath, and R. J. Plemmons (1988). "Cholesky Downdating on a Hypercube,"in G. Fox (1988), 1592–1598.

C.-T. Pan (1993). "A Perturbation Analysis of the Problem of Downdating a Cholesky Factorization," *Lin. Alg. and Its Applic.* 183, 103–115.

L. Eldén and H. Park (1994). "Perturbation Analysis for Block Downdating of a Cholesky Decomposition," *Numer. Math.* 68, 457–468.

修正和还原 ULV 分解与 URV 分解及相关的课题见:

C. H. Bischof and G. M. Shroff (1992). "On Updating Signal Subspaces," *IEEE Trans. Signal Proc.* 40, 96–105.

G. W. Stewart (1992). "An Updating Algorithm for Subspace Tracking," *IEEE Trans. Signal Proc.* 40, 1535–1541.

G. W. Stewart (1993). "Updating a Rank-Revealing ULV Decomposition," *SIAM J. Matrix Anal. Appl.* 14, 494–499.

G. W. Stewart (1994). "Updating URV Decompositions in Parallel," *Parallel Computing* 20, 151–172.

H. Park and L. Eldén (1995). "Downdating the Rank-Revealing URV Decomposition," *SIAM J. Matrix Anal. Appl.* 16, 138–155.

最后, 我们提及如下关于 SVD 修正的文章:

M. Moonen, P. Van Dooren, and J. Vandewalle (1992). "A Singular Value Decomposition Updating Algorithm," *SIAM J. matrix Anal. Appl.* 13, 1015–1038.

## 12.6 修正的及结构化的特征问题

在本节中. 我们讨论带约束的、逆形式的和结构化的特征值问题. 虽然它们并不相关, 但放在一起是为了说明如何用前述章节中基本的分解思想来解决某些特殊的特征值问题.

本节与前面的内容之依赖关系如下:

| | |
|---|---|
| 5.1 节, 5.2 节, 8.1 节, 8.3 节 | →12.6.1 节 |
| 8.1 节, 8.3 节, 9.1 节 | →12.6.2 节 |

4.7 节, 8.1 节 →12.6.3 节
5.1 节, 5.2 节, 5.4 节, 7.4 节, 8.1 节, 8.2 节, 8.3 节, 8.6 节→12.6.4 节

### 12.6.1 带约束的特征值问题

假设 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 对称, 函数 $r(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}/\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}$ 的梯度为零当且仅当 $\boldsymbol{x}$ 为 $\boldsymbol{A}$ 的特征向量. 因此, $r(\boldsymbol{x})$ 的稳定值为 $\boldsymbol{A}$ 的特征值.

在某些应用中, 需要找出 $r(x)$ 在约束 $\boldsymbol{C}^{\mathrm{T}}x = 0$ 下的稳定值. 其中 $\boldsymbol{C} \in \mathbb{R}^{n\times p}, n \geqslant p$. 假设

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{Z} = \begin{array}{c} \left[\begin{array}{cc} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{array}\right] \begin{array}{l} r \\ n-r \end{array} \\ \begin{array}{cc} r & \quad p-r \end{array} \end{array} \qquad r = \operatorname{rank}(\boldsymbol{C})$$

为 $\boldsymbol{C}$ 的完全正交分解. 定义矩阵 $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ 如下:

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{B} = \begin{array}{c} \left[\begin{array}{cc} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} \end{array}\right] \begin{array}{l} r \\ n-r \end{array} \\ \begin{array}{cc} r & \quad n-r \end{array} \end{array}$$

以及令

$$\boldsymbol{y} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x} = \left[\begin{array}{c} \boldsymbol{u} \\ \boldsymbol{v} \end{array}\right] \begin{array}{l} r \\ n-r \end{array}.$$

因为 $\boldsymbol{C}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{0}$ 变成了 $\boldsymbol{S}^{\mathrm{T}}\boldsymbol{u} = \boldsymbol{0}$. 故原问题变为在约束 $\boldsymbol{u} = \boldsymbol{0}$ 下寻找 $r(\boldsymbol{y}) = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{y}/\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}$ 的稳定值. 但这只是相当于找出 $(n-r)\times(n-r)$ 对称矩阵 $\boldsymbol{B}_{22}$ 的稳定值 (特征值).

### 12.6.2 两个逆特征值问题

考虑在上一小节中 $r = 1$ 的情形. 设 $\tilde{\lambda}_1 \leqslant \cdots \leqslant \tilde{\lambda}_{n-1}$ 为 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}/\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}$ 在约束 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} = 0$ 下的稳定值. 由定理 8.1.7 知, 这些平稳点把 $\boldsymbol{A}$ 的稳定值 $\lambda_i$ 分割开来:

$$\lambda_n \leqslant \tilde{\lambda}_{n-1} \leqslant \lambda_{n-1} \leqslant \cdots \leqslant \lambda_2 \leqslant \tilde{\lambda}_1 \leqslant \lambda_1.$$

现假设 $\boldsymbol{A}$ 的特征值互不相同, 且有已知数 $\tilde{\lambda}_1, \cdots, \tilde{\lambda}_{n-1}$ 满足:

$$\lambda_n < \tilde{\lambda}_{n-1} < \lambda_{n-1} < \cdots < \lambda_2 < \tilde{\lambda}_1 < \lambda_1.$$

我们试图找出一单位向量 $\boldsymbol{c} \in \mathbb{R}^n$, 使得 $\tilde{\lambda}_i$ 是 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$ 在约束 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x} = 1$ 和 $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} = 0$ 下的稳定值.

为了确定 $c$ 所具有的性质, 我们应用 Lagrange 乘子法. 令函数

$$\phi(\boldsymbol{x}, \lambda, \mu) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - \lambda(x^{\mathrm{T}}x - 1) + 2\mu\boldsymbol{x}^{\mathrm{T}}c$$

的梯度为零, 可得一重要方程 $(\boldsymbol{A} - \lambda\boldsymbol{I})x = -\mu c$. 因此 $(\boldsymbol{A} - \lambda\boldsymbol{I})$ 非奇且 $\boldsymbol{x} = -\mu(\boldsymbol{A} - \lambda\boldsymbol{I})^{-1}\boldsymbol{c}$, 在这个方程的两边乘以 $\boldsymbol{c}^{\mathrm{T}}$ 并利用特征值分解 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q} = \operatorname{diag}(\lambda_i)$, 可得

$$0 = \sum_{i=1}^{n} \frac{d_i^2}{\lambda_i - \lambda},$$

其中 $\boldsymbol{d}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{c}$, 即

$$p(\lambda)\equiv\sum_{i=1}^{n}d_i^2\prod_{\substack{j=1\\j\neq i}}^{n}(\lambda_j-\lambda)=0.$$

注意, $1=\|c\|_2^2=\|d\|_2^2=d_1^2+\cdots+d_n^2$ 是 $(-\lambda)^{n-1}$ 的系数. 因为 $p(\lambda)$ 是以 $\tilde{\lambda}_1,\cdots,\tilde{\lambda}_{n-1}$ 为零点的多项式, 故有

$$p(\lambda)=\prod_{j=1}^{n-1}(\tilde{\lambda}_j-\lambda).$$

从 $p(\lambda)$ 的两个表达式可知

$$d_k^2=\prod_{j=1}^{n-1}(\tilde{\lambda}_j-\lambda_k)\Big/\prod_{\substack{j=1\\j\neq k}}^{n-1}(\lambda_j-\lambda_k),\quad k=1:n. \tag{12.6.1}$$

除了符号，$d_k$ 可以确定. 这样对原问题有 $2^n$ 个不同的解 $\boldsymbol{c}=\boldsymbol{Q}\boldsymbol{d}$.

一个相关的逆特征值问题是找一个三对角矩阵

$$\boldsymbol{T}=\begin{bmatrix}\alpha_1&\beta_1&&\cdots&0\\\beta_1&\alpha_2&\ddots&&\vdots\\&\ddots&\ddots&\ddots&\\\vdots&&\ddots&\ddots&\beta_{n-1}\\0&\cdots&&\beta_{n-1}&\alpha_n\end{bmatrix}$$

使得 $\boldsymbol{T}$ 的特征值为 $\{\lambda_1,\cdots,\lambda_n\},T(2:n,2:n)$ 的特征值为 $\{\tilde{\lambda}_1,\cdots,\tilde{\lambda}_{n-1}\}$, 其中

$$\lambda_1>\tilde{\lambda}_1>\lambda_2>\cdots>\lambda_{n-1}>\tilde{\lambda}_{n-1}>\lambda_n.$$

我们说明如何用 Lanczos 方法来计算三对角矩阵 $\boldsymbol{T}$. 注意, $\tilde{\lambda}_i$ 是函数

$$\phi(y)=\frac{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}$$

在约束 $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{y}=0$ 下的平稳点, 其中 $\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_1,\cdots,\lambda_n)$, $\boldsymbol{d}$ 由 (12.6.1) 确定. 如果我们对 $\boldsymbol{A}=\boldsymbol{\Lambda}$ 和 $\boldsymbol{q}_1=\boldsymbol{d}$ 运用 Lanczos 迭代 (9.1.3), 则可产生正交矩阵 $\boldsymbol{Q}$ 和三对角矩阵 $\boldsymbol{T}$ 使得 $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{Q}=\boldsymbol{T}$. 令 $\boldsymbol{x}=\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{y}$, 可知 $\tilde{\lambda}_i$ 是

$$\psi(x)=\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{T}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}$$

在约束 $\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{x}=0$ 下的平稳点. 这正好是 $T(2:n,2:n)$ 的特征值.

### 12.6.3 Toeplitz 特征值问题

假设

$$\boldsymbol{T}=\begin{bmatrix}1&\boldsymbol{r}^{\mathrm{T}}\\\boldsymbol{r}&\boldsymbol{G}\end{bmatrix}$$

是一个对称正定的 Toepliz 矩阵, $r \in \mathbb{R}^{n-1}$. 我们的目标是在假设 $\lambda_{\min}(\boldsymbol{T}) < \lambda_{\min}(\boldsymbol{G})$ 下计算 $\boldsymbol{T}$ 的最小特征值 $\lambda_{\min}(\boldsymbol{T})$. Cybenko and Van Loan(1986) 考虑了该问题, 它在信号处理中有应用.

设

$$\begin{bmatrix} 1 & \boldsymbol{r}^{\mathrm{T}} \\ \boldsymbol{r} & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \alpha \\ y \end{bmatrix} = \lambda \begin{bmatrix} \alpha \\ y \end{bmatrix},$$

即

$$\alpha + \boldsymbol{r}^{\mathrm{T}}\boldsymbol{y} = \lambda\alpha, \quad \alpha\boldsymbol{r} + \boldsymbol{G}\boldsymbol{y} = \lambda\boldsymbol{y}.$$

若 $\lambda \notin \lambda(\boldsymbol{G})$, 则 $y = -\alpha(G - \lambda\boldsymbol{I})^{-1}r, \alpha \neq 0$ 且

$$\alpha + \boldsymbol{r}^{\mathrm{T}}[-\alpha(\boldsymbol{G} - \lambda\boldsymbol{I})^{-1}\boldsymbol{r}] = \lambda\alpha.$$

因此 $\lambda$ 是有理函数

$$f(\lambda) = 1 - \lambda - \boldsymbol{r}^{\mathrm{T}}(\boldsymbol{G} - \lambda\boldsymbol{I})^{-1}\boldsymbol{r}$$

的零点. 我们在 8.5 节和 12.1 节中处理过类似的函数. 在这种情况下, $f$ 总是有负的斜率:

$$f'(\lambda) = -1 - \|(\boldsymbol{G} - \lambda\boldsymbol{I})^{-1}\boldsymbol{r}\|_2^2 \leqslant -1.$$

如果 $\lambda < \lambda_{\min}(G)$, 则它的二阶导数也是负的:

$$f''(\lambda) = -2\boldsymbol{r}^{\mathrm{T}}(\boldsymbol{G} - \lambda\boldsymbol{I})^{-3}\boldsymbol{r} \leqslant 0.$$

利用这些事实可证, 若

$$\lambda_{\min}(\boldsymbol{T}) \leqslant \lambda^{(0)} < \lambda_{\min}(\boldsymbol{G}), \tag{12.6.2}$$

则 Newton 迭代

$$\lambda^{(k+1)} = \lambda^{(k)} - \frac{f(\lambda^{(k)})}{f'(\lambda^{(k)})} \tag{12.6.3}$$

从右边单调地收敛到 $\lambda_{\min}(\boldsymbol{T})$. 注意

$$\lambda^{(k+1)} = \lambda^{(k)} + \frac{1 + \boldsymbol{r}^{\mathrm{T}}\boldsymbol{w} - \lambda^{(k)}}{1 + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}},$$

其中 $\boldsymbol{w}$ 是带位移的 Yule-Walker 方程组

$$(\boldsymbol{G} - \lambda^{(k)}\boldsymbol{I})\boldsymbol{w} = -\boldsymbol{r}$$

之解. 因为 $\lambda^{(k)} < \lambda_{\min}(\boldsymbol{G})$, 所以 $\boldsymbol{G} - \lambda^{(k)}\boldsymbol{I}$ 是正定的. 因而, 对正规 Toeplitz 矩阵 $(\boldsymbol{G} - \lambda^{(k)}\boldsymbol{I})/(1 - \lambda^{(k)})$ 可应用算法 4.7.1.

满足 (12.6.2) 的初始值可以通过观察对矩阵 $\boldsymbol{T}_\lambda(\boldsymbol{T} - \lambda\boldsymbol{I})/(1 - \lambda)$ 的 Durbin 算法来得到. 对此矩阵, “$r$” 向量是 $r/(1-\lambda)$, 所以 Durbin 算法 (4.7.1) 变为:

$r = r/(1-\lambda)$
$y^{(1)} = -r_1$

$$
\begin{aligned}
&\textbf{for } k = 1:n-1 \\
&\quad \beta_k = 1 + [r^{(k)}]^{\mathrm{T}} y^{(k)} \\
&\quad \alpha_k = -(r_{k+1} + r^{(k)\mathrm{T}} E_k y^{(k)})/\beta_k \\
&\quad z^{(k)} = y^{(k)} + \alpha_k E_k y^{(k)} \\
&\quad y^{(k+1)} = \begin{bmatrix} z^{(k)} \\ \alpha_k \end{bmatrix} \\
&\textbf{end}
\end{aligned}
\tag{12.6.4}
$$

从 4.7.2 节的讨论知 $\beta_1, \cdots, \beta_k > 0$ 能保证 $T_\lambda(1:k+1, 1:k+1)$ 是正定的. 因此, 适当地修正后的算法 (12.6.4) 可用来计算 $m(\lambda)$, 它是使得 $\beta_1, \cdots, \beta_m$ 为正数, 但 $\beta_{m+1} \leqslant 0$ 的最大指标 $m$. 注意到, 若 $m(\lambda) = n-2$, 则 (12.6.2) 成立. 这建议如下的二分法:

$$
\begin{aligned}
&\text{选取 } L \text{ 和 } R, \text{ 使得 } L \leqslant \lambda_{\min}(T) < \lambda_{\min}(G) \leqslant R \\
&\quad \textbf{until } m = n-2 \\
&\qquad \lambda = (L+R)/2 \\
&\qquad m = m(\lambda) \\
&\qquad \textbf{if } m < n-2 \\
&\qquad\quad R = \lambda \\
&\qquad \textbf{end} \\
&\qquad \textbf{if } m = n-1 \\
&\qquad\quad L = \lambda \\
&\qquad \textbf{end} \\
&\quad \textbf{end}
\end{aligned}
\tag{12.6.5}
$$

闭区间 $[L, R]$ 始终包含使得 $m(\lambda) = n-2$ 的点 $\lambda$. 所以算法结束时, $\lambda$ 具有这一性质.

初值区间有几种可能的选择. 一种想法是令 $L = 0, R = 1 - |r_1|$, 因为

$$
0 < \lambda_{\min}(\boldsymbol{T}) < \lambda_{\min}(\boldsymbol{G}) \leqslant \lambda_{\min}\left(\begin{bmatrix} 1 & r_1 \\ r_1 & 1 \end{bmatrix}\right) = 1 - |r_1|.
$$

这里的上界由定理 8.1.7 给出.

注意, (12.6.4) 和 (12.6.5) 中的迭代步至多需要 $O(n^2)$ 个 flop. Cybenko and Van Loan (1986) 给出了只需 $O(\log n)$ 次迭代的直观证明.

### 12.6.4 正交矩阵的特征值问题

计算实正交矩阵 $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ 的特征值和特征向量是信号处理中的一个问题, 见 Cybenko(1985). $\boldsymbol{A}$ 的特征值都在单位圆上, 此外

$$
\cos(\theta) \pm \mathrm{i}\sin(\theta) \in \lambda(\boldsymbol{A}) \Leftrightarrow \cos\theta \in \lambda\left(\frac{\boldsymbol{A} + \boldsymbol{A}^{-1}}{2}\right) = \lambda\left(\frac{\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}}}{2}\right).
$$

这提示我们用 Schur 分解来计算 $\mathrm{Re}(\lambda(\boldsymbol{A}))$:

$$\boldsymbol{Q}^{\mathrm{T}}\left(\frac{\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}}}{2}\right)\boldsymbol{Q}=\mathrm{diag}(\cos(\theta_1),\cdots,\cos(\theta_n)).$$

然后用公式 $s=\sqrt{1-c^2}$ 来计算 $\mathrm{Im}(\lambda(\boldsymbol{A}))$. 不幸的是, 当 $|c|\approx 1$ 时, 由于浮点运算中有效数字的抵消, 这个公式不能给出一个准确的正弦值. 不过, 可以用反对称矩阵 $(\boldsymbol{A}-\boldsymbol{A}^{\mathrm{T}})/2$ 来计算小的 "正弦" 特征值. 但这样的话, 我们讨论的是一种需要一对满 Schur 分解的方法, 故它失去了吸引力.

Ammar, Gragg, and Reichel(1986) 给出了一种可避免这些困难的方法, 它涉及有趣的奇异值分解的应用. 我们只介绍他们的算法中有关特征值的那一部分. 算法的导出是构造性的, 因为它实际上用到了我们所学过的每一种分解.

第一步是用正交矩阵把 $\boldsymbol{A}$ 化为上 Hessenberg 矩阵, $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{Q}=\boldsymbol{H}$ (通常情况下, $\boldsymbol{A}$ 已经就是 Hessenberg 形式). 不失一般性, 可假设 $\boldsymbol{H}$ 不可约, 且它的次对角元为正数.

若 $n$ 为奇数, 则 $\boldsymbol{A}$ 一定有一实特征值, 因为实矩阵的复特征值总是以共轭对的形式出现. 在这种情形下, 仔细应用特征向量方程 $\boldsymbol{H}x=x$ 或 $\boldsymbol{H}x=-x$, 用 $O(n)$ 工作量就可把原问题降成 $n-1$ 维. 见 Gragg(1986). 因此, 我们可假定 $n$ 为偶数.

对 $1\leqslant k\leqslant n-1$, 定义反射矩阵 $\boldsymbol{G}_k\in\mathbb{R}^{n\times n}$:

$$\boldsymbol{G}_k=\boldsymbol{G}_k(\phi_k)=\begin{bmatrix}\boldsymbol{I}_{k-1} & 0 & 0 & 0\\ 0 & -c_k & s_k & 0\\ 0 & s_k & c_k & 0\\ 0 & 0 & 0 & \boldsymbol{I}_{n-k-1}\end{bmatrix}$$

其中 $c_k=\cos(\phi_k), s_k=\sin(\phi_k), 0<\phi_k<\pi$. 可以找出矩阵 $\boldsymbol{G}_1,\cdots,\boldsymbol{G}_{n-1}$, 使得

$$\boldsymbol{H}=(\boldsymbol{G}_1\cdots\boldsymbol{G}_{n-1})\mathrm{diag}(1,\cdots,1,-c_n),$$

其中 $c_n=\pm 1$. 这正是 $\boldsymbol{H}$ 的 QR 分解. 正弦值 $s_1,\cdots,s_{n-1}$ 是 $\boldsymbol{H}$ 的次对角元. "$R$" 矩阵是对角形的, 这是因为它既是正交的又是三角的. 因为反射矩阵的行列式为 $-1$. 所以 $\det(\boldsymbol{H})=c_n$, 这个数是 $\boldsymbol{H}$ 的特征值的乘积. 所以, 若 $c_n=-1$, 则 $\{-1,1\}\subseteq\lambda(\boldsymbol{H})$. 在这种情况下也可进行降阶.

所以总的说来, 我们可假设 $n$ 为偶数, 且

$$\boldsymbol{H}=\boldsymbol{G}_1(\phi_1)\cdots\boldsymbol{G}_{n-1}(\phi_{n-1})\boldsymbol{G}_n(\phi_n),$$

这里 $\boldsymbol{G}_n=\boldsymbol{G}_n(\phi_n)=\mathrm{diag}(1,\cdots 1,-c_n), c_n=1$. 要求的特征值为

$$\lambda(\boldsymbol{H})=\{\cos(\theta_k)\pm \mathrm{i}\ \sin(\theta_k)\}_{k=1}^{m},\tag{12.6.6}$$

其中 $m=n/2$.

余弦值 $c_1,\cdots,c_n$ 称为 Schur 参数. 正如我们所指出的, 相应的正弦值是 $H$ 的次对角元. 应用这些数, 可以显式地构造一对对角矩阵 $\boldsymbol{B}_C,\boldsymbol{B}_S\in\mathbb{R}^{n\times n}$ 使得

$$\sigma(\boldsymbol{B}_C(1:m,1:m)) = \{\cos(\theta_1/2), \cdots, \cos(\theta_m/2)\}, \tag{12.6.7}$$

$$\sigma(\boldsymbol{B}_S(1:m,1:m)) = \{\sin(\theta_1/2), \cdots, \sin(\theta_m/2)\}, \tag{12.6.8}$$

$B_C(1:m,1:m)$ 和 $B_S(1:m,1:m)$ 的奇异值可以用双对角 SVD 算法求出. 当 $0 < \theta_k \leqslant \pi/2$ 时, 可以从 $\sin(\theta_k/2)$ 精确求出 $\theta_k$; 当 $\pi/2 \leqslant \theta_k < \pi$ 时, 也可以从 $\cos(\theta_k/2)$ 精确求出 $Q_k$. $B_C$ 和 $B_S$ 的构造基于下面三个事实.

(1) $\boldsymbol{H}$ 相似于 $\tilde{\boldsymbol{H}} = \boldsymbol{H}_o\boldsymbol{H}_e$, 其中 $\boldsymbol{H}_o$ 和 $\boldsymbol{H}_e$ 分别是奇数个和偶数个反射矩阵的乘积:

$$\begin{aligned} \boldsymbol{H}_o &= \boldsymbol{G}_1\boldsymbol{G}_3\cdots\boldsymbol{G}_{n-1}, \\ \boldsymbol{H}_e &= \boldsymbol{G}_2\boldsymbol{G}_4\cdots\boldsymbol{G}_n, \end{aligned}$$

这些矩阵是 $2\times 2$ 块或 $1\times 1$ 块组成的块对角形矩阵, 即

$$\boldsymbol{H}_o = \mathrm{diag}(\boldsymbol{R}(\phi_1), \boldsymbol{R}(\phi_3), \cdots, \boldsymbol{R}(\phi_{n-1})), \tag{12.6.9}$$

$$\boldsymbol{H}_e = \mathrm{diag}(1, \boldsymbol{R}(\phi_2), \boldsymbol{R}(\phi_4), \cdots, \boldsymbol{R}(\phi_{n-2}), -1), \tag{12.6.10}$$

其中

$$\boldsymbol{R}(\phi) = \begin{bmatrix} -\cos(\phi) & \sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix}. \tag{12.6.11}$$

(2) 对称三对角矩阵

$$\boldsymbol{C} = \frac{\boldsymbol{H}_o + \boldsymbol{H}_e}{2} \quad \text{和} \quad \boldsymbol{S} = \frac{\boldsymbol{H}_o - \boldsymbol{H}_e}{2} \tag{12.6.12}$$

的特征值为

$$\lambda(\boldsymbol{C}) = \{\pm\cos(\theta_1/2), \cdots, \pm\cos(\theta_m/2)\}, \tag{12.6.13}$$

$$\lambda(\boldsymbol{S}) = \{\pm\sin(\theta_1/2), \cdots, \pm\sin(\theta_m/2)\}. \tag{12.6.14}$$

(3) 可以构造满足 (12.6.7) 和 (12.6.8) 的双对角矩阵:

$$\boldsymbol{U}_C^{\mathrm{T}}\boldsymbol{C}\boldsymbol{V}_C = \boldsymbol{B}_C, \quad \boldsymbol{U}_S^{\mathrm{T}}\boldsymbol{S}\boldsymbol{V}_S = \boldsymbol{B}_S.$$

变换矩阵 $\boldsymbol{U}_C, \boldsymbol{V}_C, \boldsymbol{U}_S, \boldsymbol{V}_S$ 是已知的反射矩阵 $\boldsymbol{G}_k$ 和简单的交换矩阵的乘积.

我们用证明 $\boldsymbol{H}$ 与 $\boldsymbol{H}_o\boldsymbol{H}_e$ 相似来开始这三个事实的验证. $n=8$ 的情形足以说明问题. 定义正交矩阵 $\boldsymbol{P}$:

$$\boldsymbol{P} = \boldsymbol{F}_7\boldsymbol{F}_5\boldsymbol{F}_3, \quad \text{其中} \begin{cases} \boldsymbol{F}_3 = \boldsymbol{G}_3\boldsymbol{G}_4\boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8, \\ \boldsymbol{F}_5 = \boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8, \\ \boldsymbol{F}_7 = \boldsymbol{G}_7\boldsymbol{G}_8. \end{cases}$$

既然这些反射矩阵是对称的, 且当 $|i-j| \geqslant 2$ 时, $\boldsymbol{G}_i\boldsymbol{G}_j = \boldsymbol{G}_j\boldsymbol{G}_i$, 所以有:

$$\begin{aligned} \boldsymbol{F}_3\boldsymbol{H}\boldsymbol{F}_3^{\mathrm{T}} &= (\boldsymbol{G}_3\boldsymbol{G}_4\boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8)(\boldsymbol{G}_1\boldsymbol{G}_2\boldsymbol{G}_3\boldsymbol{G}_4\boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8)(\boldsymbol{G}_3\boldsymbol{G}_4\boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8)^{\mathrm{T}} \\ &= (\boldsymbol{G}_3\boldsymbol{G}_4\boldsymbol{G}_5\boldsymbol{G}_6\boldsymbol{G}_7\boldsymbol{G}_8)\boldsymbol{G}_1\boldsymbol{G}_2 \end{aligned}$$

$$
\begin{aligned}
&= G_1G_2G_3G_4G_5G_6G_7G_8, \\
F_5(F_3HF_3^{\mathrm{T}})F_5^{\mathrm{T}} &= (G_5G_6G_7G_8)(G_1G_2G_3G_4G_5G_6G_7G_8)(G_5G_6G_7G_8)^{\mathrm{T}} \\
&= (G_5G_6G_7G_8)G_1G_3G_2G_4 \\
&= G_1G_3G_5G_2G_4G_6G_7G_8 \\
PHP^{\mathrm{T}} &= F_7(F_5F_3HF_3^{\mathrm{T}}F_5^{\mathrm{T}})F_7^{\mathrm{T}} \\
&= (G_7G_8)(G_1G_3G_5G_2G_4G_6G_7G_8)(G_7G_8)^{\mathrm{T}} \\
&= (G_1G_3G_5G_7)(G_2G_4G_6G_8) \equiv H_oH_e,
\end{aligned}
$$

我们需要建立的三个事实中的第二个反映了 $\tilde{H} = H_oH_e$ 的特征值和 (12.6.12) 中矩阵 $C$ 的特征值与矩阵 $S$ 的特征值之间的联系. 从 (12.6.9) 和 (12.6.11) 可知这些矩阵是对称的、三对角的和不可约的, 例

$$
C = \frac{1}{2}\begin{bmatrix} -c_1 & s_1 & 0 & 0 \\ s_1 & c_1 - c_2 & s_2 & 0 \\ 0 & s_2 & c_2 - c_3 & s_3 \\ 0 & 0 & s_3 & c_3 - c_4 \end{bmatrix},
$$

$$
S = \frac{1}{2}\begin{bmatrix} -c_1 & s_1 & 0 & 0 \\ s_1 & c_1 + c_2 & -s_2 & 0 \\ 0 & -s_2 & -c_2 - c_3 & s_3 \\ 0 & 0 & s_3 & c_3 + c_4 \end{bmatrix}.
$$

利用定义, 易证:

$$
\begin{aligned}
\frac{\tilde{H} + \tilde{H}^{\mathrm{T}}}{2} &= \frac{H_oH_e + (H_oH_e)^{-1}}{2} = \frac{H_oH_e + H_eH_o}{2} = 2C^2 - I, \\
\frac{\tilde{H} + \tilde{H}^{\mathrm{T}}}{2\mathrm{i}} &= \frac{H_oH_e - (H_oH_e)^{-1}}{2\mathrm{i}} = \frac{H_oH_e - H_eH_o}{2\mathrm{i}} = -2\mathrm{i}SC.
\end{aligned}
$$

这表明 $\mathrm{Re}(\lambda(\tilde{H})) = \lambda(2C^2 - I)$, $\mathrm{Im}(\lambda(\tilde{H})) = \lambda(-2\mathrm{i}CS)$. 因此, (12.6.13) 和 (12.6.14) 成立.

不把这些半角的正余弦值看成 $n \times n$ 矩阵的特征值, 而是更有效地把它们视为 $m \times m$ 矩阵的奇异值. 这使我们双对角化 $C$ 和 $S$. 完成此任务的正交等价变换基于 $H_o$ 和 $H_e$ 的 Schur 分解. (12.6.11) 定义的 $2 \times 2$ 反射矩阵 $R(\phi)$ 的特征值为 1 和 $-1$, 它的 Schur 分解如下:

$$
R(\phi/2)R(\phi)R(\phi/2) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.
$$

因此, 若

$$
\begin{aligned}
Q_o &= \mathrm{diag}(R(\phi_1/2), R(\phi_3/2), \cdots, R(\phi_{n-1}/2)), \\
Q_e &= \mathrm{diag}(1, R(\phi_2/2), R(\phi_4/2), \cdots, R(\phi_{n-2}/2), -1),
\end{aligned}
$$

则从 (12.6.9) 和 (12.6.10) 可知 $\boldsymbol{H}_o$ 和 $\boldsymbol{H}_e$ 有下列 Schur 分解:

$$\begin{aligned}\boldsymbol{Q}_o\boldsymbol{H}_o\boldsymbol{Q}_o &= \boldsymbol{D}_o = \mathrm{diag}(1,-1,1,-1,\cdots,1,-1),\\ \boldsymbol{Q}_e\boldsymbol{H}_e\boldsymbol{Q}_e &= \boldsymbol{D}_e = \mathrm{diag}(1,1,-1,1,-1,\cdots,1,-1,-1).\end{aligned}$$

矩阵

$$\begin{aligned}\boldsymbol{C}^{(1)} &= \boldsymbol{Q}_o\boldsymbol{C}\boldsymbol{Q}_e = \frac{1}{2}\boldsymbol{Q}_o(\boldsymbol{H}_o+\boldsymbol{H}_e)\boldsymbol{Q}_e = \frac{1}{2}(\boldsymbol{D}_o(\boldsymbol{Q}_o\boldsymbol{Q}_e)+(\boldsymbol{Q}_o\boldsymbol{Q}_e)\boldsymbol{D}_e),\\ \boldsymbol{S}^{(1)} &= \boldsymbol{Q}_o\boldsymbol{S}\boldsymbol{Q}_e = \frac{1}{2}\boldsymbol{Q}_o(\boldsymbol{H}_o-\boldsymbol{H}_e)\boldsymbol{Q}_e = \frac{1}{2}(\boldsymbol{D}_o(\boldsymbol{Q}_o\boldsymbol{Q}_e)-(\boldsymbol{Q}_o\boldsymbol{Q}_e)\boldsymbol{D}_e)\end{aligned}$$

分别与 $\boldsymbol{C}$ 和 $\boldsymbol{S}$ 有相同的奇异值. 下面分析它们的结构. 首先, 注意到 $\boldsymbol{Q}_o\boldsymbol{Q}_e$ 呈带状:

$$\boldsymbol{Q}_o\boldsymbol{Q}_e = \begin{bmatrix} \times & \times & \times & 0 & 0 & 0 & 0 & 0\\ \times & \times & \times & 0 & 0 & 0 & 0 & 0\\ 0 & \times & \times & \times & \times & 0 & 0 & 0\\ 0 & \times & \times & \times & \times & 0 & 0 & 0\\ 0 & 0 & 0 & \times & \times & \times & \times & 0\\ 0 & 0 & 0 & \times & \times & \times & \times & 0\\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times\\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

($n=8$ 的例子充分说明了从这一点得出的主要思想). 如果 $\boldsymbol{D}_o(i,i)$ 与 $\boldsymbol{D}_e(j,j)$ 的符号相反, 则 $\boldsymbol{C}_{ij}^{(1)}=0$, 由此可知 $\boldsymbol{C}^{(1)}$ 有如下形状:

$$\boldsymbol{C}^{(1)} = \boldsymbol{Q}_o\boldsymbol{C}\boldsymbol{Q}_e = \begin{bmatrix} a_0 & b_1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & b_2 & 0 & 0 & 0 & 0 & 0\\ 0 & a_2 & 0 & b_3 & 0 & 0 & 0 & 0\\ 0 & 0 & a_3 & 0 & b_4 & 0 & 0 & 0\\ 0 & 0 & 0 & a_4 & 0 & b_5 & 0 & 0\\ 0 & 0 & 0 & 0 & a_5 & 0 & b_6 & 0\\ 0 & 0 & 0 & 0 & 0 & a_6 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & a_7 & b_8 \end{bmatrix}.$$

类似地, 若 $\boldsymbol{D}_o(i,i)$ 与 $\boldsymbol{D}_e(j,j)$ 的符号相同, 则 $S_{ij}^{(1)}=0$, 由此可知 $\boldsymbol{S}^{(1)}$ 的形状为

$$\boldsymbol{S}^{(1)} = \boldsymbol{Q}_o\boldsymbol{S}\boldsymbol{Q}_e = \begin{bmatrix} 0 & 0 & f_1 & 0 & 0 & 0 & 0 & 0\\ e_2 & d_2 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & d_3 & 0 & f_3 & 0 & 0 & 0\\ 0 & e_4 & 0 & d_4 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & d_5 & 0 & f_5 & 0\\ 0 & 0 & 0 & e_6 & 0 & d_6 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & d_7 & f_7\\ 0 & 0 & 0 & 0 & 0 & 0 & e_8 & 0 \end{bmatrix}$$

变换这些矩阵的行和列就可得到双对角形式:

$$
\boldsymbol{B}_C = \boldsymbol{C}^{(1)}([1\ \ 3\ \ 5\ \ 7\ \ 2\ \ 4\ \ 6\ \ 8],[1\ \ 2\ \ 4\ \ 6\ \ 3\ \ 5\ \ 7\ \ 8])
= \left[\begin{array}{cccc|cccc}
a_0 & b_1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & a_2 & b_3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & a_4 & b_5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & a_6 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & b_2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & a_3 & b_4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & a_5 & b_6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & a_7 & b_8
\end{array}\right],
$$

$$
\boldsymbol{B}_S = \boldsymbol{S}^{(1)}([2\ \ 4\ \ 6\ \ 8\ \ 1\ \ 3\ \ 5\ \ 7],[1\ \ 2\ \ 4\ \ 6\ \ 3\ \ 5\ \ 7\ \ 8])
= \left[\begin{array}{cccc|cccc}
e_2 & d_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & e_4 & d_4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & e_6 & d_6 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & e_8 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & f_1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & d_3 & f_3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & d_5 & f_5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & d_7 & f_7
\end{array}\right]
$$

不难证明元素 $a, b, d, e$ 和 $f$ 均非零. 这说明 $\boldsymbol{B}_C(1:m, 1:m)$ 和 $\boldsymbol{B}_S(1:m, 1:m)$ 的奇异值均不相同. 因为

$$
\sigma(\boldsymbol{C}) = \sigma(\boldsymbol{B}_C) = \{\cos(\theta_1/2), \cos(\theta_1/2), \cdots, \cos(\theta_m/2), \cos(\theta_m/2)\},
$$

$$
\sigma(\boldsymbol{S}) = \sigma(\boldsymbol{B}_S) = \{\sin(\theta_1/2), \sin(\theta_1/2), \cdots, \sin(\theta_m/2), \sin(\theta_m/2)\},
$$

我们证实了 (12.6.7) 和 (12.6.8)

## 习　题

**12.6.1** 已知 $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, 考虑在约束:

$$
\begin{aligned}
\boldsymbol{C}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{0}, \quad & \boldsymbol{C} \in \mathbb{R}^{n\times p}, \quad n \geqslant p \\
\boldsymbol{D}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{0}, \quad & \boldsymbol{D} \in \mathbb{R}^{m\times q}, \quad m \geqslant q
\end{aligned}
$$

下求$\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{y}) = \dfrac{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}}{\|\boldsymbol{y}\|_2\|\boldsymbol{x}\|_2}$ $(\boldsymbol{y} \in \mathbb{R}^m, \boldsymbol{x} \in \mathbb{R}^n)$ 的平稳点问题. 试说明怎样通过先计算 $\boldsymbol{C}$ 和 $\boldsymbol{D}$ 的完全正交分解再计算 $\boldsymbol{A}$ 的变换矩阵之子矩阵的 SVD 来求解此问题.

**12.6.2** 设 $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{p\times n}$, 假设 $\text{rank}(A) = n$, $\text{rank}(B) = p$. 应用本节的方法, 说明如何求解问题:

$$\min_{Bx=0} \frac{\|b - Ax\|_2^2}{\|x\|_2^2 + 1} = \min_{Bx=0} \frac{\left\|[A, b]\begin{bmatrix} x \\ -1 \end{bmatrix}\right\|_2^2}{\left\|\begin{bmatrix} x \\ -1 \end{bmatrix}\right\|_2^2}.$$

证明这是一个约束 TLS 问题. 它总有解吗?

**12.6.3** 设 $A \in \mathbb{R}^{n\times n}$ 对称, $B \in \mathbb{R}^{p\times n}$ 的秩为 $p, d \in \mathbb{R}^p$. 说明如何在 $\|x\|_2 = 1$ 和 $Bx = d$ 的约束下求极小化 $x^{\mathrm{T}}Ax$ 的解. 指出何时解不存在.

**12.6.4** 设 $A \in \mathbb{R}^{n\times n}$ 是大型稀疏对称矩阵, $C \in \mathbb{R}^{n\times p}$ 也是大型稀疏的. 怎样用 Lanczos 方法求函数 $r(x) = (x^{\mathrm{T}}Ax)/(x^{\mathrm{T}}x)$ 在约束 $C^{\mathrm{T}}x = 0$ 下的平稳点. 这里假设已有稀疏的 QR 分解 $C = QR$.

**12.6.5** 试找出矩阵 $A = \begin{bmatrix} 0 & A_1 & 0 & 0 \\ 0 & 0 & A_2 & 0 \\ 0 & 0 & 0 & A_3 \\ A_4 & 0 & 0 & 0 \end{bmatrix}$ 的特征值和特征向量与矩阵 $\tilde{A} = A_1A_2A_3A_4$ 的特征值和特征向量之间的关系. 假设 $A$ 中的对角块都是方阵.

**12.6.6** 证明: 若 (12.6.2) 成立, 则 $\lambda^{(k+1)} = \lambda^{(k)} - f(\lambda^{(k)})/f'(\lambda^{(k)})$ 从右边单调地收敛于 $\lambda_{\min}(\mathrm{T})$.

**12.6.7** 回想 4.7 节的内容, 我们知可以在 $O(n^2)$ 个 flop 之内计算对称正定的 Toeplitz 矩阵的逆. 利用该事实, 找一个基于 $\|T^{-1}\|_\infty$ 和 $\|G^{-1}\|_\infty$ 的 (12.6.5) 的初始区间.

**12.6.8** 矩阵 $A \in \mathbb{R}^{n\times n}$是中心对称的, 如果它是对称和反向对称的, 即 $A = E_nAE_n$, $E_n = I_n(:, n : -1 : 1)$. 证明: 如果 $n = 2m$, 正交矩阵 $Q = \frac{1}{\sqrt{2}}\begin{bmatrix} I_m & I_m \\ E_m & -E_m \end{bmatrix}$, 则

$$Q^{\mathrm{T}}AQ = \begin{bmatrix} A_{11} + A_{12}E_m & 0 \\ 0 & A_{11} - A_{12}E_m \end{bmatrix},$$

其中$A_{11} = A(1 : m, 1 : m), A_{12} = A(1 : m, m+1 : n)$. 证明: 若 $n = 2m$ 则中心对称矩阵的 Schur 分解可只用对称矩阵的 Schur 分解所需 flop 数的 $\frac{1}{4}$ 就可实现. 在两种情况下都假定可用 QR 算法. 当 $n = 2m + 1$ 时, 情况又如何?

**12.6.9** 设 $F, G \in \mathbb{R}^{n\times n}$ 是对称的, 且 $Q = \underset{p\quad n-p}{[Q_1 \quad Q_2]}$ 是一 $n \times n$ 正交矩阵. 说明如何计算 $Q$ 和 $p$ 使得

$$f(Q, p) = \mathrm{tr}(Q_1^{\mathrm{T}}FQ_1) + \mathrm{tr}(Q_2^{\mathrm{T}}GQ_2)$$

极大化. (提示: $\mathrm{tr}(Q_1^{\mathrm{T}}FQ_1) + \mathrm{tr}(Q_2^{\mathrm{T}}GQ_2) = \mathrm{tr}(Q_1^{\mathrm{T}}(F - G)Q_1) + \mathrm{tr}(G)$.)

**12.6.10** 设 $A \in \mathbb{R}^{n\times n}$ 给定, 考虑在所有秩不超过 $r$ 的对称半正定矩阵的集合上极小化 $\|A - S\|_F$. 证明: $S = \sum_{i=1}^{\min\{k,r\}} \lambda_i q_i q_i^{\mathrm{T}}$ 是这个问题的解, 其中 $\frac{A + A^{\mathrm{T}}}{2} = Q\mathrm{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)Q^{\mathrm{T}}$ 是 $A$ 的对称部分的 Schur 分解, $Q = [q_1, \cdots, q_n]$ 且

$$\lambda_1 \geqslant \cdots\cdots \geqslant \lambda_k > 0 \geqslant \lambda_{k+1} \geqslant \cdots \geqslant \lambda_n.$$

**12.6.11** 对一般的 $n$(偶数), 证明 $\boldsymbol{H}$ 相似于 $\boldsymbol{H}_o\boldsymbol{H}_e$, 矩阵如 12.6.4 节中所定义.

**12.6.12** 证明在 12.6.4 节中的双对角矩阵 $\boldsymbol{B}_C(1:m,1:m)$ 和 $\boldsymbol{B}_S(1:m,1:m)$ 之对角元和次对角元非零. 并确定它们的值.

**12.6.13** 形如 $\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{G} \\ \boldsymbol{F} & -\boldsymbol{A}^{\mathrm{T}} \end{bmatrix}$ 的 $2n \times 2n$ 实矩阵称为 Hamilton 矩阵, 若 $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ 和 $\boldsymbol{F}, \boldsymbol{G} \in \mathbb{R}^{n\times n}$ 是对称的. 等价地, 如果定义正交矩阵

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I}_n \\ -\boldsymbol{I}_n & \boldsymbol{0} \end{bmatrix},$$

则 $\boldsymbol{M} \in \mathbb{R}^{2n\times 2n}$ 是 Hamilton 矩阵当且仅当 $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{J} = -\boldsymbol{M}^{\mathrm{T}}$. (a) 证明 Hamilton 矩阵的特征值由一负一正的对组成. (b) 矩阵 $\boldsymbol{S} \in \mathbb{R}^{2n\times 2n}$ 是辛矩阵, 若 $\boldsymbol{J}^{\mathrm{T}}\boldsymbol{S}\boldsymbol{J} = -\boldsymbol{S}^{-\mathrm{T}}$. 证明: 若 $S$ 是辛矩阵, $\boldsymbol{M}$ 是 Hamilton 矩阵, 则 $\boldsymbol{S}^{-1}\boldsymbol{M}\boldsymbol{S}$ 也是 Hamilton 矩阵. (c) 证明: 若 $\boldsymbol{Q} \in \mathbb{R}^{2n\times 2n}$ 是正交的, 且是辛矩阵, 则 $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}_1 & \boldsymbol{Q}_2 \\ -\boldsymbol{Q}_2 & \boldsymbol{Q}_1 \end{bmatrix}$, 这里 $\boldsymbol{Q}_1^{\mathrm{T}}\boldsymbol{Q}_1 + \boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{Q}_2 = I_n$ 且 $\boldsymbol{Q}_2^{\mathrm{T}}\boldsymbol{Q}_1$ 是对称的. 因此, 一个形如 $G(i, i+n, \theta)$ 的 Givens 旋转矩阵是正交的辛矩阵, $n \times n$ Householder 矩阵的直和也是如此. (d) 说明如何计算一个正交辛矩阵 $\boldsymbol{U}$ 使得

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{U} = \begin{bmatrix} \boldsymbol{H} & \boldsymbol{R} \\ \boldsymbol{D} & -\boldsymbol{H}^{\mathrm{T}} \end{bmatrix},$$

其中 $\boldsymbol{H}$ 是上 Hessenberg 矩阵, $\boldsymbol{D}$ 为对角矩阵.

## 本节注释与参考文献

12.6.1 节和 12.6.2 节中出现的逆特征值问题在下列综述文章中出现.

G. H. Golub (1973). "Some Modified Matrix Eigenvalue Problems," *SIAM Review* 15, 318–344.

D. Boley and G. H. Golub (1987). "A Survey of Matrix Inverse Eigenvalue Problems," *Inverse Problems* 3, 595–622.

平稳点问题的参考文献包括:

G. E. Forsythe and G. H. Golub (1965). "On the Stationary Values of a Second-Degree Polynomial on the Unit Sphere," *SIAM J. App. Math.* 13, 1050–1068.

G. H. Golub and R. Underwood (1970). "Stationary Values of the Ratio of Quadratic Forms Subject to Linear Constraints," *Z. Angew. Math. Phys.* 21, 318–326.

S. Leon (1994). "Maximizing Bilinear Forms Subject to Linear Constraints," *Lin. Alg, and Its Applic.* 210, 49–58.

在约束 $\boldsymbol{B}x = d$ 和 $\|x\|_2 = 1$ 下极小化 $x^{\mathrm{T}}\boldsymbol{A}x$ 的一个算法可参见:

W. Gander, G. H. Golub, and U. von Matt (1991). "A Constrained Eigenvalue Problem," in *Numerical Linear Algebra, Digital Signal Processing, and Parallel Algorithms*, G. H. Golub and P. Van Dooren (eds), Springer-Verlag, Berlin.

挑选的一些讨论逆特征值问题的文章如下：

G. H. Golub and J. H. Welsch (1969). "Calculation of Gauss Quadrature Rules," *Math. Comp.* 23, 221–230.

S. Friedland (1975). "On Inverse Multiplicative Eigenvalue Problems for Matrices," *Lin. Alg. and Its Applic.* 12, 127–138.

D. L. Boley and G. H. Golub (1978). "The Matrix Inverse Eigenvalue Problem for Periodic Jacobi Matrices," in *Proc. Fourth Symposium on Basic Problems of Numerical Mathematics*, Prague, pp. 63–76.

W. E. Ferguson (1980). "The Construction of Jacobi and Periodic Jacobi Matrices with Prescribed Spectra," *Math. Comp.* 35, 1203–1220.

J. Kautsky and G. H. Golub (1983). "On the Calculation of Jacobi Matrices," *Lin. Alg. and Its Applic.* 52/53, 439–456.

D. Boley and G. H. Golub (1984). "A Modified Method for Restructuring Periodic Jacobi Matrices." *Math. Comp.* 42, 143–150.

W. B. Gragg and W. J. Harrod (1984). "The Numerically Stable Reconstruction of Jacobi Matrices from Spectral Data," *Numer. Math*, 44, 317–336.

S. Friedland, J. Nocedal, and M. L. Overton (1987). "The Formulation and Analysis of Numerical Methods for Inverse Eigenvalue Problems," *SIAM J. Numer. Anal.* 24, 634–667.

M. T. Chu (1992). "Numerical Methods for Inverse Singular Value Problems," *SIAM J. Num. Anal.* 29, 885–903.

G. Ammar and G. He (1995). "On an Inverse Eigenvalue Problem for Unitary Matrices," *Lin. Alg. and Its Applic.* 218, 263–271.

H. Zha and Z. Zhang (1995). "A Note on Constructing a Symmetric Matrix with Specified Diagonal Entries and Eigenvalues," *BIT* 35, 448–451.

各种 Toeplitz 矩阵的特征值计算在下文给出：

G. Cybenko and C. Van Loan (1986). "Computing the Minimum Eigenvalue of a Symmetric Positive Definite Toeplitz Matrix," *SIAM J. Sci. and Stat. Comp.* 7, 123–131.

W. F. Trench (1989). "Numerical Solution of the Eigenvalue Problem for Hermitian Toeplitz Matrices," *SIAM J. Matrix Anal. Appl.* 10, 135–146.

L. Reichel and L. N. Trefethen (1992). "Eigenvalues and Pseudo-eigenvalues of Toeplitz Matrices," *Lin. Alg. and Its Applic.* 162/163/164, 153–186.

S. L. Handy and J. L. Barlow (1994). "Numerical Solution of the Eigenproblem for Banded, Symmetric Toeplitz Matrices," *SIAM J. Matrix Anal. Appl.* 15, 205–214.

酉特征值和正交特征值问题的处理见：

H. Rutishauser (1966). "Bestimmung der Eigenwerte Orthogonaler Matrizen," *Numer. Math.* 9, 104–108.

P. J. Eberlein and C. P. Huang (1975). "Global Convergence of the QR Algorithm for Unitary Matrices with Some Results for Normal Matrices," *SIAM J. Numer. Anal.* 12, 421–453.

G. Cybenko (1985). "Computing Pisarenko Frequency Estimates," in *Proceedings of the*

*Princeton Conference on Information Science and Systems*, Dept.of Electrical Engineering,Princeton University.

W. B. Gragg (1986)."The OR Algorithm for Unitary Hessenberg Matrices," *J. Comp. Appl. Math.* 16, 1–8.

G. S. Ammar, W. B. Gragg, and L. Reichel (1985)."On the Eigenproblem for Orthogonal Matrices, " *Proc. IEEE Conference on Decision and Control*, 1963–1966.

W. B. Gragg and L. Reichel (1990)."A Divide and Conquer Method for Unitary and Orthogonal Eigenproblems, "*Numer. Math.* 57, 695–718.

Hamilton 特征问题 (见习题 12.6.13) 在最优控制论中广泛出现且十分重要的.

C. C. Paige and C. Van Loan (1981)."A Schur Decomposition for Hamiltonian Matrices," *Lin. Alg. and Its Applic.* 41, 11–32.

C. Van Loan (1984)."A Symplectic Method for Approximating All the Eigenvalues of a Hamiltonian Matrix, "*Lin.Alg.and Its Applic.* 61, 233–252.

R. Byers (1986)."A Hamiltonian QR Algorithm, "*SIAM J. Sci. and Stat. Comp.* 7, 212–229.

V. Mehrmann (1988)."A Symplectic Orthogonal Method for Single Input or Single Output Discrete Time Optimal Quadratic Control Problems, "*SIAM J. Matrix Anal. Appl.* 9, 221–247.

G. Ammar and V. Mehrmann (1991)."On Hamiltonian and Symplectic Hessenberg Forms," *Lin. Alg.and Its Application* 149, 55–72.

A. Bunse-Gerstner, R. Byers, and V. Mehrmann (1992)."A Chart of Numerical Methods for Structured Eigenvalue Problems, "*SIAM J. Matrix Anal. Appl.* 13, 419–453.

其他关于修正特征值问题结构化特征值问题的文章包括:

A. Bunse-Gerstner and W. B. Gragg (1988)."Singular Value Decompositions of Complex Symmetric Matrices," *J. Comp. Applic. Math.* 21, 41–54.

R. Byers (1988)."A Bisection Method for Measuring the Distance of a Stable Matrix to the Unstable Matrices, "*SIAM J. Sci, Stat. Comp.* 9, 875–881.

J. W. Demmel and W. Gragg (1993)."On Computing Accurate Singular Values and Eigenvalues of Matrices with Acyclic Graphs," *Lin. Alg. and Its Applic.* 185, 203–217.

A. Bunse-Gerstner, R. Byers,and V. Mehrmann (1993)."Numerical Methods for Simultaneous Diagonalization," *SIAM J. Matrix Anal. Appl.* 14, 927–949.

# 索　引

**E**



**F**



**G**

**H**



**J**



**K**

**T**



**V**



**W**



**X**

**Y**



**Z**

# 矩阵计算（第3版）
# Matrix Computations

“……多年来，这本书一直是我在研究生院讲‘数值线性代数’的教材。”

——袁亚湘（中国运筹学学会理事长，冯康奖得主）

“本书内容非常丰富，有老而经典的，也有新的、正在研究中的课题。无论你是数值线性代数领域的工作人员，还是学生，这都是一本有价值的参考书。”

——SIAM Review

本书是国际上数值计算方面的权威著作，有“圣经”之称，被美国加州大学、斯坦福大学、华盛顿大学、芝加哥大学、中国科学院研究生院等很多知名学府用作相关课程的教材或主要参考书。

本书系统地介绍了矩阵计算的基本理论和方法。书中的许多算法都有现成的软件包实现，每节后还附有习题，并有注释和大量参考文献，非常有助于自学。

**Gene H. Golub** (1932－2007) 美国科学院、工程院和艺术科学院院士，世界著名的数值分析专家，现代矩阵计算的奠基人，生前曾任斯坦福大学教授。他是矩阵分解算法的主要贡献者，与William Kahan在1970年给出了奇异值分解的可行算法，该算法一直沿用至今。他发起并组织了工业与应用数学国际会议。

**Charles F. Van Loan** 著名数值分析专家。美国康奈尔大学教授，曾任该校计算机科学系主任。他于1973年在密歇根大学获得博士学位，师从Cleve Moler。

**延伸阅读**



图灵网站：www.turingbook.com 热线：(010)510951
反馈/投稿/推荐信箱：contact@turingbook.com
有奖勘误：debug@turingbook.com

**分类建议** 数学/计算数学

人民邮电出版社网址：www.ptpress.com.cn

ISBN 978-7-115-24785-8

9 787115 247858

ISBN 978-7-115-24785-8

定价：89.00元